Phylogenetic inference using Generative Adversarial Networks

Megan L. Smith[1*] and Matthew W. Hahn[1]

[1]Department of Computer Science and Department of Biology, Indiana University, Bloomington, IN 47405-7000.

*To whom correspondence should be addressed.

Abstract

Motivation: The application of machine learning approaches in phylogenetics has been impeded by the vast model space associated with inference. In particular, supervised machine learning approaches require data from across this space to train models. Because of this requirement, previous approaches have typically been limited to inferring relationships among unrooted quartets of taxa, where there are only three possible topologies. Here, we explore the potential of generative adversarial networks (GANs) to address this limitation. GANs consist of a generator and a discriminator: at each step, the generator aims to create data that is similar to real data, while the discriminator attempts to distinguish generated and real data. By using an evolutionary model as the generator, we use GANs to make evolutionary inferences, and, since a new model can be considered at each iteration, heuristic searches of complex model spaces are possible. Thus, GANs offer a potential solution to the challenges of applying machine learning to inferring phylogenetic relationships.

Results: We developed phyloGAN, a GAN that infers phylogenetic relationships among species. phyloGAN takes as input a concatenated alignment, or a set of gene alignments, and then infers a phylogenetic tree either considering or ignoring gene tree heterogeneity. phyloGAN performs well for up to ten taxa in the concatenation case, and performs well for six taxa when considering gene tree heterogeneity. Future developments to the network should aim to make the exploration of this more complex model space more efficient to reduce run times and to enable the consideration of larger trees.

Availability: phyloGAN is available on github: https://github.com/meganlsmith/phyloGAN/.

Contact: mls16@indiana.edu

1 Introduction

Inferring phylogenetic relationships among species is a central goal of many studies in evolutionary biology. Advances in sequencing technologies and in methods for inferring phylogenies from these data have proliferated in recent years (Scornavacca et al., 2020). However, phylogenetic inference is complicated by several factors, including gene tree heterogeneity due to incomplete lineage sorting (ILS), horizontal gene transfer, and gene duplication and loss (Maddison, 1997), as well as rate variation across sites and taxa. At present, most studies rely on concatenation approaches (which ignore gene tree heterogeneity) and/or summary approaches (which take inferred gene trees as input to infer a species tree). While both approaches have proven powerful, they suffer from distinct limitations. Concatenation approaches can be misled when gene tree heterogeneity is high (Kubatko and Degnan, 2007). On the other hand, summary approaches assume that gene trees have been inferred accurately, and that recombination occurs between but not within the genes used for inference (Bryant and Hahn, 2020). An ideal approach would take as input a sequence alignment and infer a species tree directly while considering processes affecting gene tree topologies, branch lengths, and sequence evolution. While such approaches exist (e.g.,*BEAST; Heled and Drummond, 2010), they are computationally demanding and therefore not applicable to large genomic datasets.

Machine learning has proven to be a powerful approach in population genetics when large datasets need to be analyzed under complex models (Schrider and Kern, 2018). In supervised machine learning, a model is trained using data with known labels. Since labeled training data are generally not available in population genetics, simulations are used to construct these training datasets. In a typical supervised machine learning approach to classification, simulations are performed under each model of interest, a classifier is trained to discriminate amongst these models using the simulated data, and, then, the classifier is applied to empirical data to select the best model. While this approach is powerful when a small number of models are considered, it requires that a large number of simulations be conducted under each model of interest prior to training the classifier. This quickly becomes computationally infeasible as model spaces become larger.

In phylogenetics, the model space is, at a minimum, the number of possible tree topologies. While there are only three possible unrooted trees for four taxa, this number sharply increases with the number of taxa: with only ten taxa there are 2,027,025 possible topologies. Because of this, supervised machine learning approaches have largely been limited to inferring relationships among quartets of taxa (e.g., Suvorov et al., 2020; Zou et al.,

2020; Solís-Lemus et al., 2022). While these approaches performed well on quartets, this limited taxon sampling prevents these methods from competing with traditional approaches (Zaharias et al., 2022). Although other approaches have been proposed to apply machine learning in phylogenetics (e.g., Nesterenko et al., 2022), no current approach can infer a species tree for more than four taxa directly from DNA sequence data using a supervised machine learning framework.

The problem of a large model space in phylogenetics is not new: likelihood, parsimony, and Bayesian approaches have all confronted this challenge by using heuristic search strategies to find the optimal tree topology without visiting all possible topologies. In a heuristic search, a new tree is proposed each iteration, and, if that tree is better than the current tree based on some optimality criterion (e.g., has a higher likelihood), then the tree is accepted. The obvious path forward to applying deep learning approaches in phylogenetics is to employ similar heuristic search strategies. One candidate for this is generative adversarial networks (GANs; Goodfellow et al, 2014).

GANs have previously been applied in population genetics to simulate artificial genomes (Yelmen et al., 2021; Booker et al., 2022) and to estimate population genetic parameters (Wang et al., 2021). GANs consist of two networks: a generator and a discriminator. The goal of the generator is to generate data that are indistinguishable from observed data, while the goal of the discriminator is to distinguish real from generated data. At the end of the training process, the generator should produce realistic datasets. Recently, a GAN was used to estimate parameters of a population genetic model (Wang et al., 2021). The generator simulated data under some parameterized population genetic model, and at each iteration, new parameter values were proposed via simulated annealing. When the GAN produced realistic data and training was halted, the current parameters of the generator were inferred to be the best parameters to explain the real data. Using a similar approach, GANs could be used to heuristically search tree space.

Here, we introduce a new model, phyloGAN. phyloGAN heuristically explores phylogenetic tree space to find a tree topology that produces generated data that are similar to observed data. We introduce two versions of phyloGAN: a concatenated version and a version that accommodates gene tree heterogeneity (such as is produced by ILS). We demonstrate phyloGAN on several simulated datasets and show that it performs well in the concatenation case for up to fifteen taxa and can accurately infer relationships among six taxa while considering gene tree heterogeneity. We also apply phyloGAN to infer relationships among seven species of fungi.

2 Methods

Given a tree topology and distribution of branch lengths, phyloGAN works by generating sequence data, comparing these data to real data using a convolutional neural network (CNN), and proposing new tree topologies to find a topology that produces data that are similar to the real data (Figure 1; Algorithm 1). We begin with a pre-GAN phase to infer a scale parameter for an exponential distribution on branch lengths that produces generated data with a number of variable sites similar to that in the real data. Then, we can start to train phyloGAN. The generator generates tree topologies that are used as input in an evolutionary simulator, and the discriminator is a CNN that learns to distinguish generated from real data.

Here, we first describe the general strategy employed in phyloGAN. We then describe our pre-GAN phase to infer a reasonable parameterization on the distribution of branch lengths. Following this, we describe the generator and discriminator in more detail. Finally, we describe an extension of phyloGAN to accommodate gene tree heterogeneity.

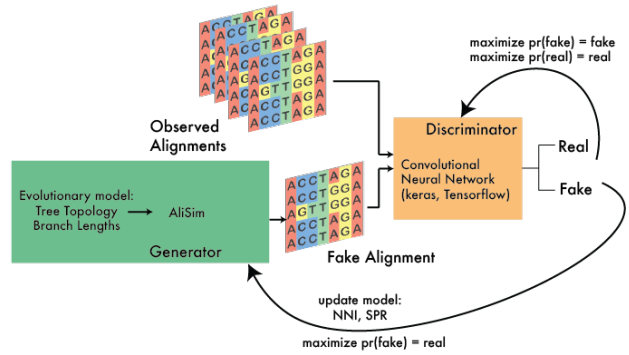

Figure 1. Overview of phyloGAN. The generator generates a tree topology and branch lengths, which are used as input into an evolutionary simulator. The discriminator is a CNN trained to differentiate real and generated data.

The main input to phyloGAN is a multiple sequence alignment. Additionally, the user must provide a model of sequence evolution and bounds on the scale parameter for the exponential distribution on branch lengths. The user must also supply several arguments related to training (described in the online documentation at https://github.com/meganlsmith/phyloGAN/). The main outputs of phyloGAN are the tree topology, τ, at each iteration, the generator and discriminator loss at each iteration, and the trained discriminator.

Training a GAN is a minimax game in which the generator aims to maximize the probability that generated data is classified as real data, and the discriminator aims to minimize the probability that either data type is misclassified. Here, our generator aims to minimize the generator loss function. As in Wang et al. (2021), with M regions of simulated data $\{z^{(1)}...z^{(m)}\}$ generated under τ, the generator loss is calculated using the cross-entropy function:

$$L_G(\tau) = -\frac{1}{M}\sum_{m=1}^{M} logD(z^{(m)}).$$

The discriminator aims to classify generated regions as fake and real regions as real by minimizing the discriminator loss function for M regions of simulated data $\{z^{(1)}, ..., z^{(m)}\}$ and M regions of real data $\{x^{(1)}, ..., x^{(m)}\}$. Again, as in Wang et al. (2021), the discriminator loss function is calculated using the binary cross-entropy function:

$$L_D(\tau) = -\frac{1}{M}\sum_{m=1}^{M}[logD\left(x^{(m)}\right) + \log\left(1 - D\left(z^{(m)}\right)\right)].$$

2.1 Stage 1: Inferring λ

To ensure that simulated datasets are reasonably similar to the real data in terms of the number of variable sites, we precede our training by inferring a speciation rate, λ, that produces datasets similar to the real data. We begin with some value of λ drawn from a uniform prior with bounds provided by the user. Then, at each training iteration we propose k new values of λ using simulated annealing. For each proposal, we simulate M regions of sequence data of length L under a random tree topology with branch lengths drawn from an exponential distribution with scale parameter $1/\lambda$. We calculate the average number of invariable sites across the M regions, and then calculate the distance between this value and the same value calculated from M regions of real data:

$$dist = \left|\frac{\sum_N pinv_{simulated}}{N} - \frac{\sum_N pinv_{empirical}}{N}\right|.$$

At each iteration, we keep the proposal with the smallest distance. Then, this distance is compared to the current distance. We accept the proposed value if it decreases the distance, or with some probability proportional to the ratio of the distances and the temperature *T*. Here, *T* determines how often we accept bad proposals. *T* starts at 1 and decreases to 0 over a fixed number of iterations. The inferred value of λ is used to parameterize the distribution of branch lengths during the training stage of phyloGAN. Specifically, during training, branch lengths are drawn from an exponential distribution with scale parameter 1/ λ.

2.2 The Generator

phyloGAN needs to generate tree topologies, τ, and branch lengths that can then be used as input into an evolutionary simulator. Here, we use nearest neighbor interchange (NNI) and subtree pruning and regrafting (SPR) moves to propose tree topologies. These moves are implemented using the python packages ETE3 (Huerta-Cepas et al., 2016) and Bio.Phylo (Talevich et al., 2012). Following these moves, AliSim (Ly-Trong et al., 2022) is used to generate sequence data under the proposed topologies. The proposed moves are determined by the temperature, *T*. As above, *T* starts at 1 and decreases to 0 over a fixed number of iterations. When *T* is near 1, SPR moves are more likely, and as *T* approaches 0, NNI moves become more likely.

Training begins either with a random start tree or with a neighbor-joining tree inferred from the real data. At each iteration, *k* new topologies are proposed using NNI and/or SPR. The generator loss is calculated for each proposed topology (with branch lengths drawn from an exponential distribution separately for each region), and the proposed topology with the lowest generator loss is retained. At the end of each iteration, the best proposed topology, τ, is always accepted if the generator loss is less than or equal to the previous iteration. Otherwise, we use the temperature *T* to determine whether to accept the proposed topology. The acceptance probability is calculated as follows:

$$p_{accepted} = \frac{L_G(\tau)}{L_G(\tau^{(i)})} * T^{(i)} * 0.1,$$

where $\tau^{(i)}$ is the proposed topology, τ is the current topology, and $T^{(i)}$ is the current temperature. If the proposed topology is accepted, the discriminator is trained as described below.

We also note one major difference between our approach and most GANs. We recalculate the generator loss at the end of each iteration (i.e., after discriminator training). This prevents phyloGAN from getting stuck when the discriminator improves substantially between iterations.

2.3 The Discriminator

As the discriminator, we use a CNN implemented in keras (Chollet et al., 2015) and tensorflow (Abadi et al., 2015), with an architecture inspired by Wang et al. (2021) and Suvorov et al. (2020). Sequence data are converted to 2D NumPy arrays (Harris et al., 2020) by coding {A,T,C,G} as {0,1,2,3}. We determine the maximum number of variable sites (*S*) to include when training by sampling *M* regions of length *L* from the real data and calculating the average number of variable sites. Any simulated or real regions with more than *S* sites are truncated to *S* sites, and any regions with fewer than *S* sites are padded with the missing data value "4." To ensure that all possible combinations of individuals are considered by the convolutions when constructing these arrays for training the discriminator, we sample all quartets of individuals and provide sets of rows for each quartet. For example, if there are five species in the dataset, the initial alignment consists of five rows. However, the 2D array consists 20 rows, corresponding to all combinations of four taxa. Training datasets are created from the real dataset in a similar fashion; *M* regions of length *L* are sampled and converted to 2D NumPy arrays as above.

The discriminator consists of several layers. First, we use a convolutional layer with 10 filters with kernel size (4,1) and a stride size (4,1) with ReLU activation. Next, we apply a MaxPooling layer with pool size (1,2) and stride size (1,2). This is followed by a second convolutional layer with 10 filters, kernel size (2,1), and stride size (4,1), which is followed by another MaxPooling layer identical to the first. Then, we apply a flattening layer, a dense layer with 128 neurons, a dropout layer with a dropout rate of 0.2, another dense layer with 128 neurons, another dropout layer, and, finally, a dense layer with a single layer and linear activation, which predicts class probabilities. When training the discriminator, we use a user-specified number of epochs. For each epoch, we simulate *M* regions under the current tree topology (with branch lengths drawn from an exponential distribution separately for each region), and then train the discriminator using a gradient descent approach.

Following Wang et al. (2021) we precede training with a pre-training phase, so that the discriminator has an overall sense of the data. For a user-specified number of pre-training iterations, we generate a random tree topology under a birth-death model with branch lengths drawn from an exponential distribution with scale parameter 1/ λ. We then generate *M* regions under the tree topology and branch lengths, draw *M* regions from the real data, and pre-train the discriminator.

2.4 Extension to gene tree heterogeneity

To extend phyloGAN to consider gene tree heterogeneity, we made several modifications. First, in stage 1, we jointly infer the coalescent time and the scaling parameter, such that we are explicitly modeling gene tree heterogeneity due to ILS. Specifically, branch lengths are now parameterized as node ages drawn from an exponential distribution with the scale parameter equal to the coalescence time, and this scale parameter is inferred in a pre-GAN phase. Additionally, there is a scaling parameter associated with simulating sequence data along these gene trees that is jointly inferred in the pre-GAN phase. Finally, in order to incorporate ILS we added a step to the simulator. For each simulated region, *M*, gene trees, *G*, are simulated within the species tree under the multispecies coalescent model using DendroPy (Sukumaran and Holder, 2010); sequence data (with alignment lengths based on the empirical data) are simulated along

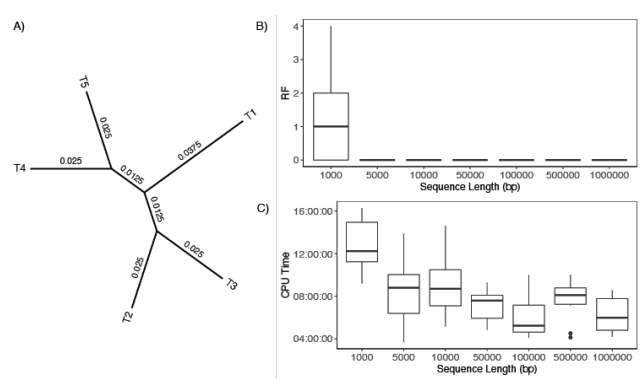

Figure 2: Results across sequence lengths. A) Five-taxon tree used in simulations. B) Robinson-Foulds distance between true tree and maximum clade credibility tree. C) CPU time of phyloGAN.

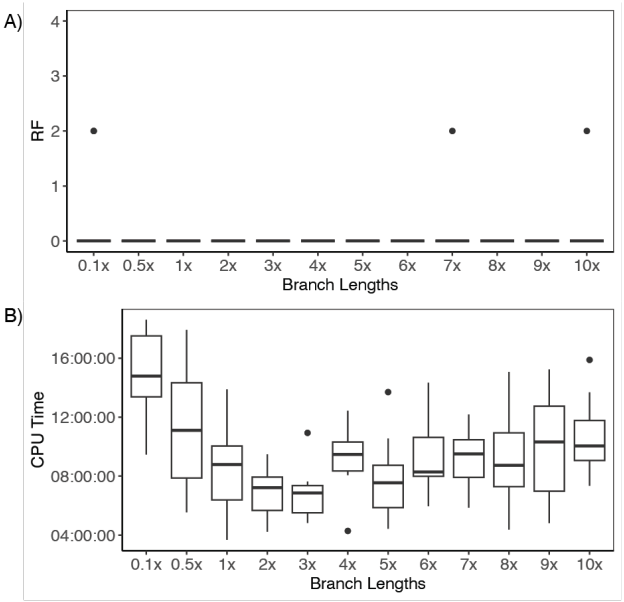

Figure 3: Results across branch lengths. A) Robinson-Foulds distance between true tree and maximum clade credibility tree. B) CPU time of phyloGAN.

these gene trees in AliSim (Ly-Trong et al., 2022), as above. The *G* matrices are concatenated into a single matrix before being used as input into the discriminator. We refer to this version as phyloGAN-ILS hereafter.

2.5 Simulations

To evaluate the performance of phyloGAN, we used several simulated datasets. For each combination of tree topology, sequence length, and branch lengths, we simulated ten alignments and ran phyloGAN on each alignment. All simulations of sequence data were performed using the Jukes-Cantor model. First, to evaluate phyloGAN's performance over a range of sequence and branch lengths, we focused on a five-taxon tree (Figure 2). As a baseline, we set all branch lengths to range between 0.0125 and 0.0375, and evaluated performance on datasets with sequence lengths ranging from 1,000 to 1,000,000 bp. Then, we explored branch lengths from 0.1x to 10x the original values with 5,000 bp sequences. To explore the performance of phyloGAN on larger trees, we considered trees with 5-20 taxa (Figure S1), with 50,000 bp sequences and branch lengths similar to the baseline case above.

We ran phyloGAN on the five-taxon datasets as follows. We considered 25 regions of 500 bp in each minibatch. We set the minimum and maximum bounds on λ to 1 and 30, respectively. Stage 1 (i.e., inferring λ) included 100 iterations and considered ten proposals per iteration. We ran 20 pre-training iterations, and trained the discriminator for 30 epochs

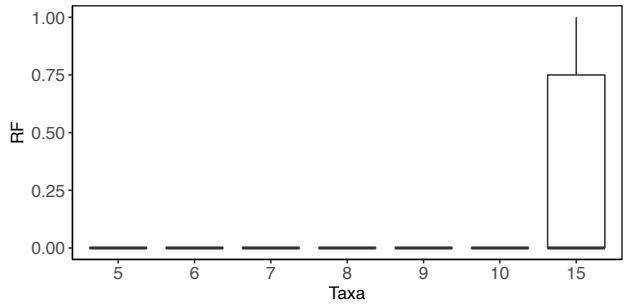

Figure 4. Results on larger trees. Normalized Robinson-Foulds (RF) distances between the true tree and the maximum clade credibility tree.

during pre-training. We then ran phyloGAN for 500 training iterations, and trained the discriminator for 25 epochs when new topologies were accepted. At each training iteration, we considered a maximum of 10 trees. For larger trees, we increased the number of training iterations to 5,000. We found the maximum clade credibility tree using the "sumtrees" command in DendroPy (Sukumaran and Holder, 2010; Sukumaran and Holder, 2015) after discarding the first 25% of trees as burn-in, and we compared this tree to the true tree to evaluate the performance of phyloGAN.

To evaluate the performance of phyloGAN-ILS, we simulated 5000 gene trees on a six-taxon species tree (Figure S2). For each gene tree, we simulated between 300 and 1200 bp of sequence data. We ran phyloGAN-ILS for 20 pretraining and 1,000 training iterations. We trained the discriminator for 20 epochs in pre-training and 30 epochs in training, as above. For each generated region, we simulated gene trees and sequence data (with lengths based on the empirical data) until we had 1000 bp of generated data. We used a mini-batch size of 25 regions.

2.6 Empirical data

To evaluate the performance of phyloGAN on an empirical dataset, we used a subset of the fungal dataset from Rasmussen and Kellis (2011). We considered seven species: *Candida lusitaniae, C. guillermondii, Debaryomyces hansenii, C. albicans, C. tropicalis, C. parapsilosis,* and *Lodderomyces elongisporus*. We retained only those gene families that were present in exactly one copy in each sampled species, and we removed all sites with > 50% gaps in a gene alignment prior to concatenating the retained gene alignments. We ran phyloGAN as described above for larger trees, and found the maximum clade credibility tree using DendroPy.

3 Results

phyloGAN was highly accurate in the five-taxon case (Figure 2). Across all sequence lengths, except 1,000 bp with the baseline branch lengths, the maximum clade credibility tree was identical to the true tree. phyloGAN generally took longer in cases where inference was more difficult (e.g., with shorter alignments; Figure 2). phyloGAN required an average of ~7 CPU hours to run with 5,000 bp, and used ~500 Mb of memory. phyloGAN was also highly accurate across all branch lengths considered (Figure 3A), and, again, CPU time tended to scale with difficulty (Figure 3B). phyloGAN performed well with as many as ten taxa, but accuracy began to decrease with fifteen taxa (Figure 4), and the CPU time scaled with the number of taxa. phyloGAN-ILS inferred the correct species tree in 100% of runs, but was substantially slower than the concatenated version (average wall time ~ 7.3 days). The maximum clade credibility tree inferred using phyloGAN for the yeast species differed from the generally accepted species tree in its placement of *D. hansenii* sister to *C. lusitaniae*, rather than *C. guillermondii* (Figure 5).

We also evaluated several metrics that could be useful in diagnosing when phyloGAN runs are complete, including generator and discriminator loss and accuracy. In an ideal scenario, generator loss should decrease throughout a GAN run, and discriminator loss should increase to a point. However, this is not the trend we see with phyloGAN. Generally, generator loss is highly variable and increases slightly early in the run. As the run continues, the loss levels out. Discriminator loss remains relatively steady throughout the run, suggesting that the discriminator learns quickly to discriminate real from generated data (Figure 6A). Ideally, discriminator accuracy should increase early in the run and then decrease to ~0.5, since generated data are becoming more similar to real data; however, with phyloGAN, discriminator accuracy increases quickly and remains high

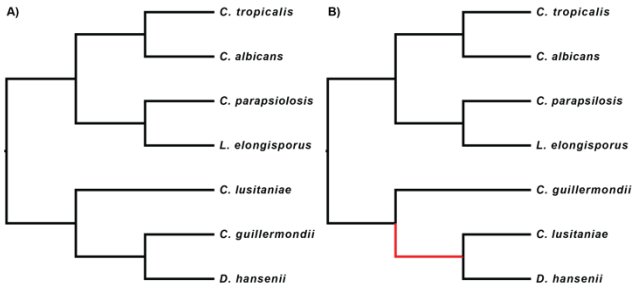

Figure 5: Results from fungal dataset. A) Tree topology from Rasmussen and Kellis (2011). B) Maximum clade credibility tree estimated in phyloGAN.

throughout the run. Generator accuracy, similarly, increases quickly and remains high throughout the run (Figure 6B).

4 Discussion

The possibility of using GANs to heuristically explore model spaces has previously been attempted in population genetic inference with respect to continuous parameters (Wang et al., 2021). Here, we take this approach to explore the discrete tree space associated with phylogenetic inference. phyloGAN offers a substantial improvement over previous machine learning approaches in phylogenetics in its ability to infer relationships for more than four taxa: phyloGAN not only ran, but performed accurately with as many as ten taxa. It performs well in relatively straightforward scenarios, but struggles both in terms of accuracy and run times more as model spaces become more complex (e.g., with larger trees). Additionally, phyloGAN is far less computationally efficient than approaches that perform similarly or better in these same scenarios (e.g., maximum likelihood). While this suggests that phyloGAN is not currently a competitive tool for phylogenetic inference, these results demonstrate that it is possible to design a GAN that efficiently explores phylogenetic model spaces. Future implementations may find ways to explore these spaces even more efficiently.

In particular, the poor performance of traditional metrics for assessing loss and accuracy in our GAN suggests that learning rates may not be optimized. We explored a number of approaches to improve these metrics, including changing the number pretraining iterations, the number of epochs used in training, and the architecture of the discriminator. However, we failed to improve performance. In an ideal GAN, both generator and discriminator accuracy would be near 0.5 by the end of the run—i.e., it should be difficult for the discriminator to distinguish real from generated data—but both metrics are near 1 in phyloGAN. One explanation for this could be our approach to branch lengths. We do not infer branch lengths, but rather integrate over branch lengths. Therefore, the real data is always a single dataset generated under a specified set of branch lengths, while the generated data are generated under a tree topology and several sampled branch lengths. This may facilitate discrimination between the real and generated datasets, and prevent accuracy from decreasing to 0.5. We also note that we recalculate the generator loss after training the discriminator at each iteration. We found that this improved performance, but it could complicate standard interpretations of loss and accuracy metrics. In the traditional approach (i.e., when loss is not recalculated after training) generator loss will most often decrease when a new proposal is accepted. However, we found that this caused phyloGAN to get stuck because discriminator performance would improve substantially between iterations, leading to challenges in beating the previous loss. Recalculating this loss prevented phyloGAN from getting stuck in this discrete model space, but may necessitate different approaches to assessing run performance.

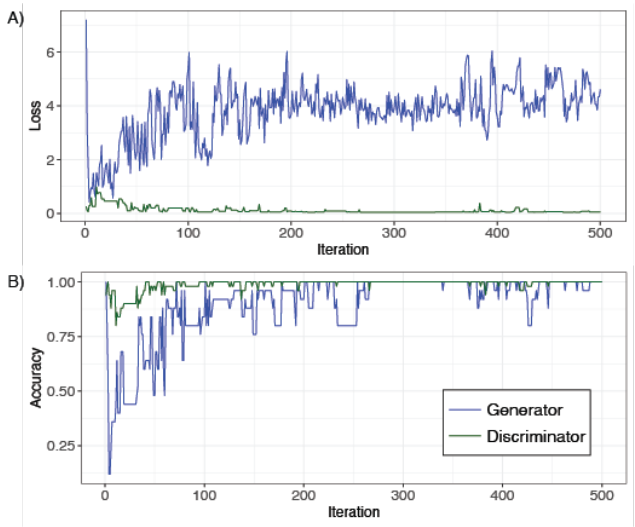

Figure 6. Performance metrics for phyloGAN. A) Generator and discriminator loss across 500 iterations in the baseline (1X branch lengths and 5,000 bp). B) Generator and discriminator accuracy under the same conditions as A.

Here, we focused on inferring a species tree directly from sequence data. While this approach allows us to avoid many of the limitations associated with summary approaches (i.e., approaches that infer a species tree from gene trees), inference could be improved by incorporating both sequence data and inferred gene trees. Recent studies have explored approaches for encoding phylogenetic trees as data in machine learning frameworks (Voznica et al., 2022; Sophia et al., 2022). Future explorations of GANs in phylogenetics may benefit from considering gene tree structures in lieu of or in addition to sequence data.

While we are cautiously optimistic about the use of phyloGAN in empirical applications, our work certainly demonstrates the potential of machine learning in phylogenetic contexts. Future work exploring alternative data structures and alternative GAN architectures (or other machine learning architectures) may pave the way to more accurate and efficient GANs for phylogenetics. Additionally, GANs are extremely flexible and can consider any process under which we can generate data. Thus, future work may explore the use of GANs while considering other processes that complicate phylogenetic inference, including both gene duplication and loss and introgression.

Acknowledgements

We thank Daniel Schrider and members of the Schrider lab for helpful discussion.

Funding

This work has been supported by a National Science Foundation postdoctoral fellowship to MLS (DBI-2009989) and an NSF grant to MWH (DEB-1936187).

Conflict of Interest: none declared.

References

Abadi, M., *et al.* (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Booker, W.W. *et al.* (2022) This population doesn't exist: learning the distribution of evolutionary histories with generative adversarial networks. *bioRxiv*, DOI:10.1101/2022.09.17.508.

Bryant, D. and Hahn, M.W. (2020) The concatenation question. In Scornavacca, C. et al. (ed.), *Phylogenetics in the genomic era*, Open access book. https://hal.inria.fr/PGE/

Chollet, F., *et al.* (2015) Keras. Software available from https://github.com/fchollet/keras.

Goodfellow, I.J. *et al.* (2014) Generative Adversarial Networks. *arXiv,* DOI: 10.48550/arxiv.1406.2661.

Harris, C.R. *et al.* (2020) Array programming with NumPy. *Nature,* 585(7825), 357-362.

Heled, J. and Drummond A.J. (2010) Bayesian inference of species trees from multilocus data. *Molecular Biology and Evolution*, 27(3), 570-580.

Huerta-Cepas, J. *et al.* (2016) ETE 3: Reconstruction, analysis, and visualization of phylogenomic data. *Molecular Biology and Evolution,* 33(6), 1635-1638.

Kubatko, L.S. and Degnan, J.H. (2007) Inconsistency of phylogenetic estimates from concatenated data under coalescence. *Systematic Biology*, 56(1), 17-24.

Ly-Trong, N., *et al.* (2022) AliSim: A fast and versatile phylogenetic sequence simulator for the genomic era. *Molecular Biology and Evolution,* 39(5), msac092.

Maddison, W.P. (1997) Gene trees in species trees. *Systematic Biology*, 46(3), 523-526.

Nesterenko, L. *et al.* (2022) Phyloformer: towards fast and accurate phylogeny estimation with self-attention networks. *bioRxiv*, DOI: 10.1101/2022.06.24.496975.

Rasmussen, M.D. and Kellis, M. (2011) A Bayesian approach for fast and accurate gene tree reconstruction. *Molecular Biology and Evolution*, 28(1), 273-290.

Schrider, D.R. and Kern, A.D. (2018) Supervised machine learning for population genetics: a new paradigm. *Trends in Genetics*, 34(4), 301-312.

Scornavacca, C. *et al.* (2020) Phylogenetics in the genomic era. Open access book. https://hal.inria.fr/PGE/

Solís-Lemus, C. *et al.* (2022) Accurate phylogenetic inference with a symmetry-preserving neural network model. *arXiv*, 2201.04663.

Sophia, L. *et al.* (2022) Deep learning from phylogenies for diversification analysis. *bioRxiv*, DOI: 10.1101/2022.09.27.509667.

Sukumaran, J. and Holder, M.T. (2010) DendroPy: A python library for phylogenetic computing. *Bioinformatics*, 26(12), 1569-1571.

Sukumaran,J. and Holder,M.T. (2015) Sumtrees: phylogenetic tree summarization v.4.5.2. Available at https://github.com/jeetsukumaran/DendroPy.

Suvorov, A. *et al.* (2020) Accurate inference of tree topologies from multiple sequence alignments using deep learning. *Systematic Biology*, 69(2), 221-233.

Talevich, E. *et al.* (2012) Bio.Phylo: A unified toolkit for processing, analyzing, and visualizing phylogenetic trees in Biopython. *BMC Bioinformatics*, 13(1), 209.

Voznica, J. *et al.* (2022) Deep learning from phylogenies to uncover the epidemiological dynamics of outbreaks. *Nature Communications*, 13(1), 3896.

Wang, Z. *et al.* (2021) Automatic inference of demographic parameters using generative adversarial networks. *Molecular Ecology Resources,* 21(8), 2689-2705.

Yelmen, B. *et al.* (2021) Creating artificial human genomes using generative neural networks. *PLoS Genetics*, 17(2), e1009303.

Zaharias, P. *et al.* (2022) Re-evaluating deep neural networks for phylogeny estimation: the issue of taxon sampling. *Journal of Computational Biology*, 29(1): 74-89.

Zou, Z. *et al.* (2020) Deep residual neural networks resolve quartet molecular phylogenies. *Molecular Biology and Evolution*, 37(5), 1495-1507.