Simulations of sequence evolution: how (un)realistic they really are and why

Johanna Trost[1,*], Julia Haag[2,*], Dimitri Höhler[2,*], Luca Nesterenko[1], Laurent Jacob[3], Alexandros Stamatakis[4,2,5], and Bastien Boussau[1]

[1]Biometry and Evolutionary Biology laboratory (LBBE) University Claude Bernard Lyon 1, Lyon, France
[2]Computational Molecular Evolution Group, Heidelberg Institute for Theoretical Studies, Heidelberg, Germany
[3]Sorbonne Université, CNRS, IBPS, Laboratory of Computational and Quantitative Biology (LCQB), UMR 7238, Paris 75005, France
[4]Biodiversity Computing Group, Institute of Computer Science, Foundation for Research and Technology - Hellas, Heraklion, Crete, Greece
[5]Institute for Theoretical Informatics, Karlsruhe Institute of Technology, Karlsruhe, Germany
[*]These authors contributed equally to this work.

July 11, 2023

Abstract

Motivation: Simulating sequence evolution plays an important role in the development and evaluation of phylogenetic inference tools. Naturally, the simulated data needs to be as realistic as possible to be indicative of the performance of the developed tools on empirical data. Over the years, numerous phylogenetic sequence simulators, employing various models of evolution, have been published with the goal to simulate such empirical-like data. In this study, we simulated DNA and protein Multiple Sequence Alignments (MSAs) under increasingly complex models of evolution with and without insertion/deletion (indel) events using a state-of-the-art sequence simulator. We assessed their realism by quantifying how well supervised learning methods are able to predict whether a given MSA is simulated or empirical.

Results: Our results show that we can distinguish between empirical and simulated MSAs with high accuracy using two distinct and independently developed classification approaches across all tested models of sequence evolution. Our findings suggest that the current state-of-the-art models fail to accurately replicate the process of evolution.

Data and Code Availability: All simulated and empirical MSAs, as well as all analysis results, are available at https://cme.h-its.org/exelixis/material/simulation_study.tar.gz. All scripts required to reproduce our results are available at https://github.com/tschuelia/SimulationStudy and https://github.com/JohannaTrost/seqsharp.

Contact: julia.haag@h-its.org

1 Introduction

Reconstructing the evolutionary history of species or genes by inferring phylogenetic trees is a ubiquitous task in comparative genomics. Typically, phylogenetic inference is based on an MSA that contains aligned sequences of the species under study. A plethora of inference algorithms, tools, and models have been developed to infer phylogenetic trees based on the MSA, for example RAxML-NG [28], IQ-Tree [33], BEAST [9], or RevBayes [23]. With recent advances in machine learning and deep learning, researchers have also started applying these techniques to phylogenetic inference problems.

For example, Abadi et al. [1] use Random Forests to predict the optimal model for branch length estimation, and Nesterenko et al. [34] use a transformer-based network architecture to predict evolutionary distances between pairs of sequences. When developing novel methods and validating their performance, comparing them to existing state-of-the-art methods on both empirical and simulated data is mandatory. Simulated data are particularly useful when a known, ground truth phylogeny is needed for conducting inference accuracy assessments. Additionally, simulated data play an important role in ABC, machine learning and deep learning methods, either for inference or training [13, 34].

In all these cases, simulated data need to be similar to empirical data: performance on simulated data is indicative of performance on empirical data only if the two are sufficiently similar. Otherwise, reconstruction methods trained on simulations may not perform analogously on empirical data. Simulated MSAs are typically obtained by generating MSAs starting from existing or simulated phylogenetic trees and a pre-specified model of evolution. Phylogenetic trees can be obtained via simulations or from various empirical databases. Models of sequence evolution either focus on substitutions, or deal with both substitution and insertions/deletions (indel) events. Models of substitution are typically from the GTR family of models for DNA sequences, or based on published substitution matrices and mixture models for protein data. Finally, models that include indel events have parameters that affect the frequency, position, and length of insertions and deletions.

Throughout the years, a plethora of data simulation tools have been developed [32, 16, 12]. Recent tools, such as the AliSim sequence simulator [32], are feature-rich and support a wide range of evolutionary models and simulation parameters.

Authors using simulated data in their publications typically set simulation parameters according to attributes (e.g., MSA lengths, or proportions of gaps) of empirical reference MSAs (see e.g., Price et al. [39]). Some also attempt to extract or sample simulation parameters from Maximum Likelihood estimates in large scale empirical databases, such as TreeBASE [37], in the hope that thereby simulated data will better resemble empirical data [1, 22]. Despite the effort, there still exist performance or program behavior differences on simulated versus empirical data. For example, Guindon et al. [19] conclude that comparing methods using simulated data is not sufficient, as "the likelihood landscape tends to be smoother than with real data", and Hoehler et al. [22] noticed differences between empirical and simulated data when comparing ML phylogenetic inference methods, concluding that there exist not yet understood differences between simulated and empirical data.

In this paper, we show that we can distinguish simulated from empirical data with up to 99% classification accuracy, depending on the used simulation model. We present two different and independently developed machine learning approaches exploiting distinct MSA characteristics for this classification task: One using Gradient Boosted Trees (GBT), and another approach based on a Convolutional Neural Network (CNN). We show that prediction accuracy decreases the more complex the model of evolution used in simulations becomes, although we also observe exceptions to this trend. Yet, for the most complex models in our experimental setup, the prediction accuracy is still very high, with the CNN based classifier achieving prediction accuracies ≥ 0.93 on all tested models. This indicates that simulated alignments are easy to distinguish from empirical alignments, as they do not appear to reproduce some characteristic features of empirical MSAs. We further show that simulating indels remains a challenging task, as including indels results in higher classification accuracies with the CNN classifiers compared to simulations without indels. Further, based on the feature importances of the GBT classifiers, we show that simulated data have more evenly distributed site substitutions than empirical data.

2 Methods

The goal of our study was to be able to distinguish between empirical and simulated DNA and protein data with high accuracy under increasingly complex models of sequence evolution. Figure 1 depicts our experimental setup for one exemplary set of empirical MSAs (empirical data collection) and one exemplary model of evolution. Using the empirical data collection and the given model of evolution, we simulated a new set of MSAs (simulated data collection) using the AliSim sequence simulator [32]. Based on the empirical and simulated data collections, we completely independently trained two distinct classifiers for each simulated data collection: a Gradient Boosted Tree (GBT) and a Convolutional Neural Network (CNN). For a specific, given MSA, each classifier predicts whether it is empirical or simulated. We assessed the performance of both classifiers via the balanced accuracy

metric (BACC) [11] and a 10-fold cross validation procedure. In the following section, we describe our experimental setup, the sequence simulation process, and both classification methods in more detail.

2.1 Alignment Simulations

For our study, we separately considered DNA and protein data. We simulated 15 MSA sets, seven sets containing DNA MSAs and eight containing protein MSAs, respectively. In the following, we refer to an MSA set as a *data collection*. To simulate the MSAs for each data collection, as well as for data discrimination, we used two empirical data collections as reference, one per data type. The empirical DNA data collection contains MSAs obtained from TreeBASE [37]. The empirical protein data collection consists of MSAs obtained from the HOGENOM database [36]. We removed outliers based on MSA length (i.e., number of sites), number of sequences, as well as MSAs with less than four sequences to ensure a reliable and efficient analysis. Very long sequences would inflate the memory footprint of the CNN, while very short MSAs are uncommon and are more difficult to accurately classify as empirical or simulated. Removing outliers in terms of sequence allowed us to deploy a balanced and representative data collection that facilitates robust and unbiased predictions.

Moreover, empirical MSAs may contain sites with ambiguous or exceptional AA/DNA codes, that are 'B', 'Z', 'J', 'U', 'O' and 'X' for protein MSAs, and 'N', 'D', 'H', 'V', 'B', 'R', 'Y', 'K', 'M', 'S', 'W' and 'X' for DNA MSAs. As a further pre-processing step, yet exclusively for the CNN classifier, we removed all MSA sites containing at least one ambiguous letter, as they would bias the prediction. For protein data this concerned 912 out of 6969 MSAs, and we removed 1.34% of all sites within these MSAs. Furthermore, 13.24% of sites in 6117 MSAs with DNA sequences were removed.

For each data type, we generated simulated data collections based on the corresponding empirical data collection. We simulated data using the AliSim sequence simulator [32] under a plethora of evolutionary models ranging from easy to complex, in terms of number of free parameters and computational methods used to derive respective amino acid (AA) substitution models. The goal of this setup was to progressively increase simulation realism. First, we simulated five DNA and seven protein data collections without gaps, which allowed us to characterize the realism of substitution models per se. To this end, we removed all sites containing gaps ('-') from all empirical MSAs. The resulting empirical data collections contain 7637 DNA MSAs and 6971 protein MSAs, respectively. We henceforth refer to these data collections as *gapless* data collections. Second, we simulated two DNA and one protein data collection with indel events, based on the empirical MSAs containing gaps (9460 DNA MSAs and 6971 protein MSAs).

In the following, we describe the simulation procedures for both data types, as well as our approach to simulate indel events, in more detail.

2.1.1 DNA Simulation

We simulated seven DNA data collections in total (5 gapless and 2 with simulated indel events), with each data collection being simulated separately under a different evolutionary model with increasing model complexity. We used the following models of evolution. As the simplest model, we used the Jukes-Cantor (JC) model (equal substitution rates and equal base frequencies) [24]. We also used the HKY model (four degrees of freedom) [21], and the General Time Reversible (GTR) model (eight degrees of freedom) [47]. To account for among site rate heterogeneity, we additionally simulated under GTR in conjunction with the Γ model [52] using four discrete rates (GTR+G). The most complex model of evolution we used for simulation was the GTR+G model, with an additional free parameter to accommodate the proportion of invariant sites (GTR+G+I) [44].

We selected 9460 empirical MSAs (*Set1*) from TreeBASE [37, 50] as basis for our simulations. We removed all sites containing gaps ('-') or fully undetermined characters ('N') from the MSAs of *Set1*. Thereby, we obtained 7637 non-empty MSAs (i.e., MSAs that still contained at least one site), which we defined as *Set2*. This lead to an MSA length reduction of around 55% compared to *Set1*. We based our five simulated DNA data collections without indel events on Set2, and the two data collections with indels on Set1.

AliSim simulates sequences along a given phylogenetic tree. We avoided the problem of simulating realistic phylogenetic trees for this purpose by running 20 independent tree searches (with 10 parsimony and 10 random starting trees) using RAxML-NG [28], for every MSA of Set2 under each of the five evolutionary models (JC, HKY, GTR, GTR+G, GTR+G+I). We then used the inferred phylogeny

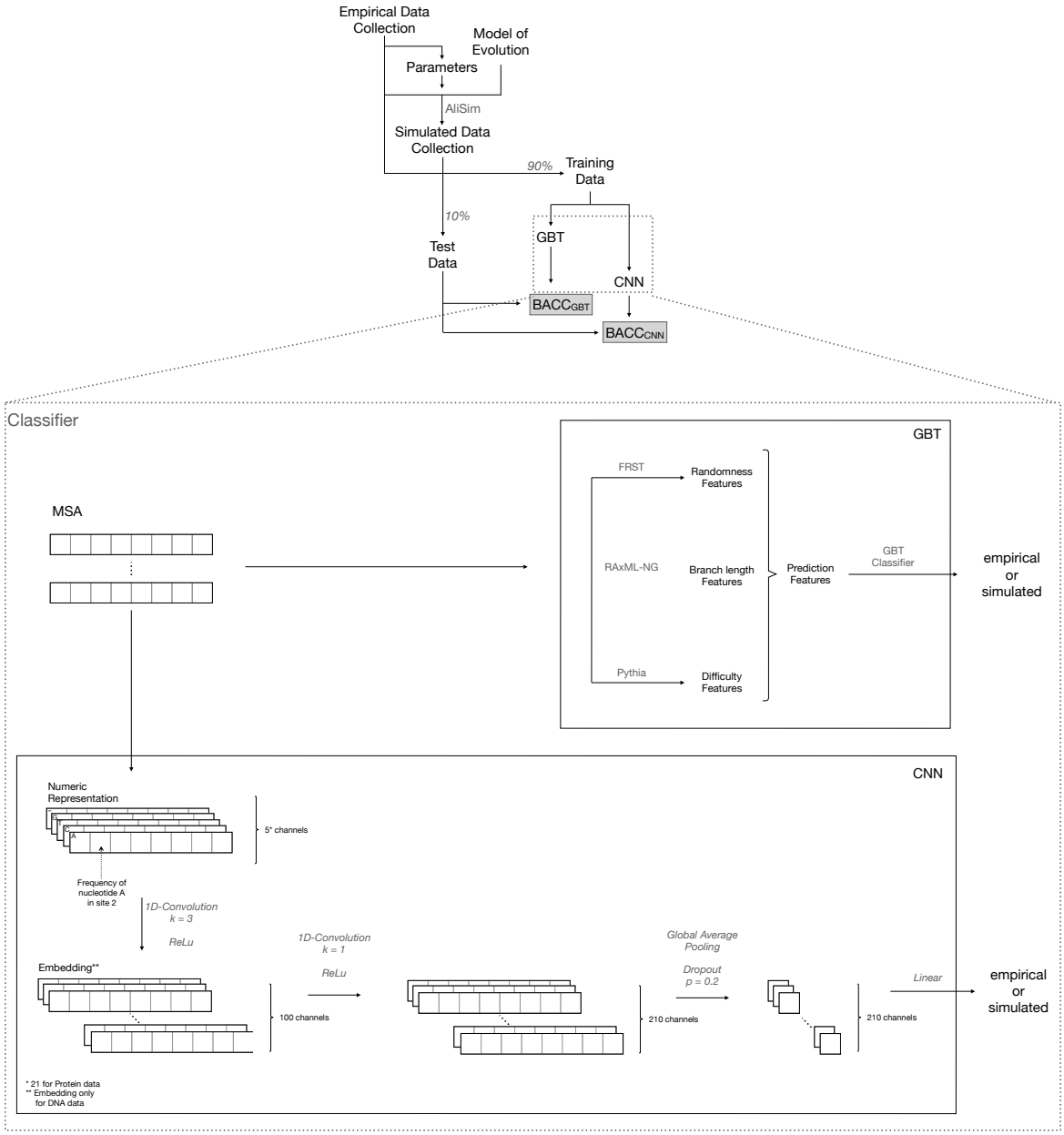


Figure 1: Schematic overview of our experimental setup. Based on a set of empirical MSAs (empirical data collection), we determined parameters for sequence simulation and simulated new MSAs (simulated data collection) under a specific model of evolution using AliSim. Using the empirical and simulated data collections, we trained two distinct classifiers: a Gradient Boosted Tree (GBT) and a Convolutional Neural Network (CNN). The goal of both classifiers is to distinguish empirical from simulated MSAs. For training and evaluating our classifiers, we used a 10-fold cross validation procedure (not depicted for simplicity). In each fold, 90% of the data were used for training and 10% were used for performance evaluation. We evaluated the overall performance of the classifiers via balanced accuracy (BACC).

and estimated model parameters to simulate MSAs using AliSim [32] based on every MSA of Set2, without specifying an indel model. In the following analyses, we refer to the resulting five gapless data collections as JC, HKY, GTR, GTR+G, and GTR+G+I according to the model of evolution used. In Section 2.1.3 below, we describe the simulation of the two additional DNA data collections with indel events.

2.1.2 Protein Simulation

We simulated seven protein data collections limited to substitution events only, and one additional data collection *with* indels.

The most rudimentary evolutionary model we used is the *Poisson* model, with equal exchangeabilities and equal stationary frequencies. We further used two empirical substitution models: the *WAG* [51] and the *LG* [30] model. The LG model is expected to produce more realistic simulations than the WAG model as the former was derived from a larger and more diverse data collection, using more refined inference techniques than the latter. Because these substitution models use a single set of stationary frequencies (i.e., one amino acid profile) to simulate all sites in an MSA, we also use evolutionary models that incorporate heterogeneity among sites by employing multiple profiles. Here, to simulate a site, a profile is drawn from a set of profiles.

To this end, we used the following two models: the C60 model with 60 profiles (*LG+C60*) [45] and the more recent UDM model with 256 profiles (*LG+S256*) [41]. The advantage of the latter model is that each profile is assigned a probability (i.e., weight) of generating a site, while under the C60 model profiles are drawn with equal probabilities. In addition, the UDM model is based on a subset of MSAs from the HOGENOM database, and should therefore generate alignments that are similar to empirical HOGENOM MSAs. To increase model complexity, we performed further simulations accounting for among site heterogeneity using the Γ model [52] again. Here, we simulated two data collections, one using four discrete Γ rate categories (*LG+S256+G4*) and the second applying a continuous Γ distribution (*LG+S256+GC*).

We set the α shape parameters of the Γ distributions based on the values inferred during tree reconstruction when building the HOGENOM database (see Supplementary Material Section 1.1.1). In the following, we will refer to these parameters as *empirical α parameters*. For the simulations, we drew α parameters from the probability density function (PDF) estimate of the empirical α parameters (see Supplementary Material Section 1.1.1). We compared the ECDF of the empirical α parameters with the ECDF of 7000 samples from the PDF estimation to guarantee a reasonable approximation of the distribution (see Figure S1).

We sampled the MSA length we used for MSA simulations from the approximated empirical distribution of HOGENOM MSA lengths, using the same approach as for the α parameters outlined in Section 1.1.1. Respective PDF and ECDF functions can also be found in the Supplementary Material in Figure S1).

We simulated all seven data collections along phylogenetic trees that were reconstructed from empirical HOGENOM MSAs where sites containing indels were removed. We performed the inferences with RAxML-NG (see Section 2.1.1).

In analogy to the simulated DNA data collections, we refer to the simulated protein data collections according to the model of evolution used. The gapless protein data collections are Poisson, WAG, LG, LG+C60, LG+S256, LG+S256+G4, and LG+S256+GC. In the following section, we describe the simulation procedure for the additional data collection with indels.

2.1.3 Simulating Indels

In addition to the gapless data collections, we simulated two DNA, and one protein data collections *with* indels. For both data types, we use the most complex models of evolution as a basis (GTR+G+I for DNA, LG+S256+GC for protein).

To generate the first DNA data collection with indels, we ran 20 independent tree searches under the GTR+G+I model for each MSA of DNA *Set1*. We then simulated MSAs with indels using two distinct procedures to generate two distinct data collections. For the first data collection, we simulated the MSAs in the same way as for the gapless collections. Then, we superimposed the gap pattern of the MSAs used as the basis of the simulation onto the simulated MSAs. We refer to this procedure as the *mimick* procedure and denote the resulting data collection as GTR+G+I+mimick.

For the second data collection, as well as the protein data collection with indels, we simulated the MSAs not only by using the inferred trees and estimated evolutionary model parameters, but by also specifying indel parameters. In the following, we describe the procedure to infer and validate these parameters. It was performed for both DNA and protein data collections separately. We refer to this procedure as the *sparta* procedure. We first used the SpartaABC tool [31] to obtain indel-specific parameters from a subset of empirical MSAs. Here, we employed the rich indel model (RIM), which differentiates between insertion and deletion events using five free parameters. The inferred parameters are: Insertion and deletion rate (I_R, D_R), root length (RL), and the parameter a that controls the Zipfian distribution of insertion and deletion lengths (A_I, A_D). We will henceforth refer to this set of parameters as *empirical indel parameters*.

To then simulate MSAs, we drew indel parameters from the joint parameter distribution of empirical indel parameters. To approximate the probability density function (PDF), Gaussian kernels were applied to the five principal components of the indel parameters. This choice was based on our observation that a more accurate match is achieved between the empirical parameters' empirical cumulative distribution function (ECDF) and the resulting parameters' ECDF when using the principal components. For the Gaussian kernels, the bandwidth was determined using Scott's rule of thumb [42]. Moreover, we employed the kernel-density estimation implementation by Virtanen et al. [49], although it tends to overestimate the distribution's actual edges. To mitigate this issue, we re-sampled values if they fell outside the bounds of the parameter prior bounds chosen by Loewenthal et al. [31]. To validate our approach, we compared the ECDF of the empirical parameter values with the ECDF of parameters sampled from the empirical PDF for each indel parameter type. Plots of the ECDFs and density functions are provided in the Supplementary Material Figure S2 and Figure S4. Moreover, we compared the density functions of empirical and simulated MSA lengths as a sanity check (see Figure S3 and Figure S5 in the Supplementary Material).

We denote the resulting DNA data collection as GTR+G+I+sparta, and the resulting protein data collection LG+S256+GC+sparta.

2.2 Classification Methods

To distinguish simulated and empirical MSAs, we developed two distinct approaches. One approach is a classical machine learning algorithm based on hand-crafted features and Gradient Boosted Trees (GBT). Using GBTs allows us to attain insights on feature importance, explain the classification results, and determine short-comings of MSA simulations. Our second approach uses Convolutional Neural Networks (CNN). In contrast to GBT, CNNs only require minimal data processing as they are able to automatically learn relevant features through training. However, to interpret these features, additional analysis is necessary. In the following, we describe our training setup for both approaches.

2.2.1 Gradient Boosted Trees

Gradient Boosted Trees (GBT) is an ensemble machine learning technique that combines multiple decision trees to obtain an accurate prediction model [17]. Training a GBT classifier consists of M sequential stages, with each stage contributing an additional decision tree that improves the estimator of the previous stage. For our experiments, we used the GBT classifier as implemented in the LightGBM framework [25].

Prediction Features To classify MSAs into simulated or empirical ones, we computed 23 features for each MSA.

Four of these features are attributes of the MSA: the *sites-over-taxa ratio*, the *patterns-over-taxa ratio*, the *patterns-over-sites ratio* and the proportion of invariant sites (*% invariant*). For data collections simulating indel events, we also used the proportion of gaps as feature (*% gaps*). Further, we quantified the signal in the MSA using the Shannon entropy [43] as average over all column entropies (*Entropy*), two metrics based on the number and frequency of patterns in the MSA (*Bollback multinomial* [8]; *Pattern entropy*), and the difficulty of the respective phylogenetic analysis as predicted by Pythia [20] (*difficulty*). In order to assess downstream effects on tree inferences using simulated and empirical data, we inferred 100 trees based on the fast-to-compute maximum parsimony criterion [14, 15] and a single Maximum Likelihood (ML) tree using RAxML-NG [28]. We added two features based on the inferred 100 maximum parsimony trees: the average pairwise topological distance using

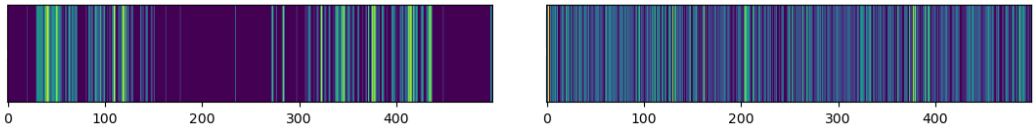


Figure 2: Visualized substitution rates for an anecdotal (specifically selected to highlight the issue) gapless empirical DNA MSA (left), and gapless simulated MSA (right) generated based on the inferred tree and estimated evolutionary model parameters of the left MSA under the GTR model. The x-axis denotes the alignment site index. A brighter color denotes a higher number of substitutions.

the Robinson-Foulds distance metric (*parsimony RF-Distance*) [40], as well as the proportion of unique topologies (*% parsimony unique*). We further refer to these features as difficulty features. Based on the ML tree inferred by RAxML-NG, we computed a set of branch length features, namely the minimum, maximum, average, standard deviation, median, and sum of all branch lengths in the ML tree ($brlen_{min}$, $brlen_{max}$, $brlen_{avg}$, $brlen_{std}$, $brlen_{med}$, $brlen_{sum}$).

We used the next six features to highlight one of the recurrent problems of simulated sequence generators: A common simplification used in generators is the assumption that substitutions occur at uniformly distributed random locations in the sequence, which appears to not be the case in real-world genetic data [10]. Thus, we expected empirical MSAs to be less uniform than simulated MSAs, and we henceforth attempted to confirm this hypothesis.

To quantify substitution frequency distributions along an MSA, we first inferred a parsimony tree using RAxML-NG. Then, based on the parsimony criterion, we calculated the number of substitutions for every site, resulting in a vector m.

Given the vectors m for empirical and simulated MSAs, we can anecdotally observe, that the locations of substitution occurrences appear to be less uniformly distributed in empirical MSAs (see Figure 2, more examples available in the Supplement) and appear more "random". To the best of our knowledge, there is no panacea in quantifying this randomness, and it is part of ongoing research in the field of cryptography. We resorted to the Fourmilab Random Sequence Tester (FRST) (https://www.fourmilab.ch/random/), that is used to evaluate pseudo-random number generators, to quantify randomness in m. FRST computes six measures of randomness: Entropy ($Entropy_{rand}$), maximum compression size reduction in percent (*comp*), Chi-Square test (Chi^2), arithmetic mean ($mean_{rand}$), Monte Carlo Value for Pi (*mcpi*) (see Section 3.2 in Supplementary Material), and Serial Correlation Coefficient (*SCC*) [27]. We executed FRST with a binary representation of m on all data collections, then we normalized the computed measures of randomness, and used these values in our predictions. We henceforth refer to this set of six features as *randomness features*.

Training and Optimization For each of the simulated data collections presented above, we trained a distinct binary GBT classifier. We trained each GBT classifier using a stratified 10-fold cross-validation procedure. The training data consisted of one simulated data collection and the empirical data collection for the respective data type. During each iteration of the cross-validation, the training data were split into 10 subsets, 9 of these 10 subsets are used to train the GBT classifier and 1 subset is used as a test set to measure the classification accuracy on data not seen during training. We report the accuracy as average accuracy on the test set across the 10 cross-validation iterations. We report the balanced accuracy (BACC) [11] as this metric allows for varying proportions of simulated/empirical MSAs in the data collection and better represents classification accuracy for imbalanced datasets. The best BACC value is 1 and the worst value is 0. We used the hyperparameter optimization framework Optuna [4] to determine the optimal set of hyperparameters for each classifier. For each GBT classifier, we performed 100 Optuna iterations using a Tree-structured Parzen Estimator algorithm [7] to sample the hyperparameter space. To prevent the classifiers from overfitting the data, based on preliminary experiments, we limited the depth of the individual decision trees to a maximum of 10, the maximum number of leaves to 20, and the minimal number of samples per leaf to 30. Additionally, we applied L1 and L2 regularization to prevent overfitting and better generalize to unseen data [18]. We determined the optimal weights of L1 and L2 regularization independently using Optuna. L1 regularization penalizes trees with many branching events, L2 penalizes large weights. A more detailed description of the feature generation, and training process, the hyperparameter optimization, as well as the hardware

setup are available in the Supplementary Material Section 3.1.

2.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a popular prediction method originally developed for computer vision and image processing. Recently, they have been applied to predict properties of biological sequences [6, 5, 53]. A CNN jointly learns a representation of the data (through convolution layer(s)) and the classification of the data based on these representations (in our case using a fully connected layer). More precisely, a convolution layer slides short probabilistic sequence motifs along the sequence, and outputs an activation profile (i.e., feature map) for each of these motifs. Here, we used a CNN to classify empirical and simulated MSAs. In the following we will detail the input to the network, its architecture, training and optimization, and the evaluation of its performance.

MSA Representation In order to obtain a numeric representation of an MSA, where the network is invariant to the order of sequences, we decided to represent the MSA using its site-wise AA or nucleotide composition, i.e., the AA or nucleotide proportions per site, which sum to one. Here, each AA/nucleotide, as well as gaps are passed to the convolution network as input features (i.e., channels), resulting in 5 (4 DNA sites + gap) or 21 (20 amino acids + gap) channels. This approach is analogous to using color channels in an image. It maintains the identity of a nucleotide/AA and is common practice when applying CNNs to biological sequences [6]. The input size was the maximum MSA length in the simulated and empirical data collection. All MSAs with fewer sites were zero-padded at their edges in order to match the fixed input size.

Empirical protein sequences typically start with Methionine (M), which simulations do not account for. We removed the first and second sites from the empirical protein data to avoid biasing the prediction. We tested the trained network on empirical validation data, including the second site, to ensure that keeping it would not have led to significantly different results. The absolute accuracy difference between data with and without the second site was below 0.0005 (see Table S2 in the Supplementary Material).

CNN Architecture We developed two architectures, one for each data type (DNA and protein). We explored alternative architectures and chose the architecture with the best balance between complexity and performance. For protein MSAs, we used a single one-dimensional convolution layer with 210 filters of size 1×21 (i.e., kernel size $\times$ input channels). Of note, these filters simply capture AA profiles, as opposed to larger motifs spanning several contiguous sites typically used in CNNs. For DNA sequences, we used a two-layer CNN, whose first layer has 100 filters of size 3×5 and is meant to capture codons. The second layer has 210 filters of size 1×100. A standard Rectified Linear Unit (ReLU) activation function is employed in both architectures [3]. For both DNA and protein architectures, the layers following convolution comprise a dropout layer with a probability of 0.2 and global average pooling averaging along the sequences. A final fully-connected layer combines all features (i.e., channels) for the binary prediction. For this, a Sigmoid activation function is used. In total, the protein network counts 4831 learnable parameters, while the DNA network has 23021 due to the additional convolution layer.

Training and Optimization To update the network weights, we employed the Adam optimizer [26] along with a binary cross-entropy loss function. The optimized parameters include the learning rate and the number of filters. For the former, we chose the learning rate range independently for each data collection and fold using the learning rate range test (LRRT) [46] on learning rates from 10^{-7} to 0.5. We then evaluated different learning rates by means of the validation loss after 100 epochs and considered the learning curve, that is, the validation and training loss over epochs. Through this procedure, we found a learning rate of 0.001 for all DNA data collections. In the case of protein data, we determined varying learning rates ranging from 0.001 to 0.0117 across different data collections. Here, the optimized learning rates correspond to one-tenth of the upper bound identified by the LRRT, which proved to be the most effective choice for the protein data collections. We used a batch size of 128 for protein data and 64 for DNA data. Furthermore, we varied the number of filters and chose the amount that yielded the maximal validation BACC. In addition to the validation BACC, we considered the Class MAE, which is the mean absolute difference between the accuracy on simulated

and empirical data collections across folds, as well as the SE, which denotes the standard error of the obtained validation BACC across folds. If these measures were strikingly large, we interpreted this as an indicator that the network needs to be improved to generalize better. As with the other classifier, we used 10-fold cross-validation. We applied an early stopping rule [38] to automatically terminate the training of every fold individually. However, for certain data collections, we observed that the chosen stopping rule seemed overly strict. The visualized learning curves indicated that the network had converged, even though the stopping criterion was not met. Consequently, we decided to manually terminate the training in these cases. Learning curves, Class MAE, and the SE can be found in the Supplementary Material (Table S1, Figure S6, Figure S7).

To compare the performance of CNNs trained on various simulated data collections, we determined the maximum validation BACC over training epochs for each CNN and for each fold. What is referred to as BACC in the following is the average BACC across folds at the selected epochs. Because we are using the same validation data to choose the stopping epoch and assess the resulting accuracy, there is a risk that this accuracy is overoptimistic. To quantify this risk, we computed summary statistics of BACCs of six to twelve epochs surrounding the selected epoch (denoted n-BACC) (see Table S1 in the Supplementary Material). All aspects of our classifier, including data preprocessing, model architecture, as well as training and evaluation were implemented in Python [48]. We employed Pytorch [35], a popular open-source machine learning framework, to implement our model and trained it on an Nvidia Tesla V100 GPU.

2.2.3 Performance Evaluation

Using the BACC metric per data collection, we compared the performance of pairs of classifiers of simulated data collections. In order to evaluate whether the difference of the BACCs of two data collections and therefore two different evolutionary models is significant, we conducted multiple unpaired two-samples t-tests, where one sample consists of the validation BACC for each fold. This allowed us to make informed suggestions regarding the evolutionary model's efficacy in producing simulations that are more or less or equally realistic. For protein data, we compared the BACCs of the following groups: Poisson vs. WAG, WAG vs. LG, LG vs. LG+C60, and all pair-wise combinations of site heterogeneous models. The null hypothesis is that these models yield equal average BACCs across folds. We rejected the null hypothesis if the resulting p-value was below the significance level of 0.05. For DNA data, we compared the BACCs of JC vs. HKY, HKY vs. GTR, GTR vs. GTR+G and GTR+G vs. GTR+G+I. To account for multiple testing, we applied Bonferroni correction, i.e., we multiplied each p-value by the number of tests for each data type separately [2]. An overview of all tests is provided in the Supplementary Material (Table S7, Table S8, and Table S9).

3 Results

Table 1 shows the BACC for our GBT and CNN classifiers across all data collections. The BACC is a measure of classification accuracy that represents the average of the fraction of correct predictions per class, which is further averaged across folds of the 10-fold cross-validation.

Both classifiers were able to accurately distinguish simulated from empirical data. The GBT classifiers achieved high BACCs for all simulated protein data collections (≥ 0.98), as well as for all gapless DNA data collections (≥ 0.89). We observed the worst BACC of 0.77 for the DNA data collection simulated under GTR+G+I with gaps simulated according to the *mimick* procedure. The CNN classifiers achieve BACCs ranging from 0.93 to 0.9996. Interestingly, the GBT classifiers show similar BACCs or even outperform the CNN on the protein data collections, but achieved lower BACCs on DNA collections.

On DNA data collections, substitution models with fewer degrees of freedom than the GTR model, namely JC and HKY, were classified more accurately (BACC=0.99 for CNN and BACC=0.96 for GBT). However, increases in model complexity did not always translate into improvements in the realism of the data. For instance, the performance of the CNN was marginally better on simulations under the HKY model than on simulations under the simpler JC model (P=0.03, see Table S1). The GBT predictions, which were equally accurate for JC and HKY simulations (BACC=0.96), did not reflect any improvement in the simulations due to more degrees of freedom in the HKY model either. Moreover, the CNN yielded the lowest BACC (0.93) on simulations conducted under the GTR

model. In contrast, simulations that included rate heterogeneity (GTR+G) were slightly easier to classify (BACC=0.94, P=0.04). Contrary to our expectations, including a proportion of invariant sites (GTR+G+I) did not result in a lower BACC compared to GTR+G simulations (BACC=0.94, P=1.0 for CNN, BACC=0.89, P=1.0 for GBT).

We did not observe the expected trend of an increased realism with an increase in model complexity for the protein data collections. For instance, the CNN had the lowest BACC on simulations under the LG substitution model (BACC=0.95) opposed to the expected mixture models. For the GBT, distinguishing the LG+S256+G4 data collection appeared to be easier than the data collection based on the simpler LG+C60 model (P=0.77). Unexpectedly, all simulations using a mixture of stationary frequency profiles (i.e., LG+C60, LG+S256, LG+S256+G4 and LG+S256+GC) were nearly perfectly discriminated from the empirical data collection with both GBT and CNN (BACC $\geq$ 0.98). With the CNN, we did not find a significant performance difference between these evolutionary models (P$\geq$0.38, see Table S9). To rule out the possibility that these rather unexpected findings are a consequence of specific behaviors inherent to the AliSim simulator, we conducted an experiment to evaluate the performance of the CNN classifier pre-trained with LG+S256 simulations on data generated using an in-house developed simulator that employs the same model. Our results showed that the CNN classifier performed comparably well on the alternative simulations (BACC=0.99). In addition, we tested the same CNN on simulations using 4096 profiles. These simulations were only slightly harder to classify (BACC=0.98) than the ones based on only 256 profiles (BACC=0.995).

The CNN trained on empirical data collections with indels and simulations under the most complex evolution model with indels (i.e., LG+S256+GC+sparta, GTR+G+I+mimick, GTR+G+I+sparta) also yielded highly accurate predictions (BACC=0.996 for protein and BACC>0.97 for DNA data). The results were similar to or better than the results obtained without indels. There was no significant difference between CNN performance on the two DNA indel models employed (P=1.0). Simulating indels increased the GBT classification accuracy for protein data (BACC=0.99) and the *sparta* based DNA data collection (GTR+G+I+sparta; BACC=0.94) compared to the same model of evolution without indel simulations (LG+S256+GC BACC=0.98; GTR+G+I BACC=0.89). We did, however, observe a significant decrease in accuracy comparing the two DNA indel models (P=0.0). GBT classified the GTR+G+I+sparta data collection with high accuracy (BACC=0.94), but showed an unexpectedly low BACC of 0.77 for GTR+G+I+mimick.

Data collection	BACC	
	GBT	CNN
DNA data collections		
JC	0.96	0.99
HKY	0.96	0.99
GTR	0.94	0.93
GTR+G	0.89	0.94
GTR+G+I	0.89	0.94
GTR+G+I+mimick	0.77	0.97
GTR+G+I+sparta	0.94	0.97
Protein data collections		
Poisson	0.99	0.9996
WAG	0.99	0.97
LG	0.99	0.95
LG+C60	0.98	0.99
LG+S256	0.99	0.995
LG+S256+G4	0.99	0.99
LG+S256+GC	0.98	0.99
LG+S256+GC+sparta	0.99	0.996

Table 1: Average of the BACC on empirical and simulated data collections across 10 folds for the GBT and CNN classifiers. Parameter configurations of simulations listed in the first column are sorted with increasing complexity from top to bottom for both DNA and protein data. For both, the last row(s) shows results on data collections with indels.

In order to gain insights into why the general classification task achieved high prediction accuracy and appears to be rather easy in general, we assessed the influence of the described features on the prediction of the GBT classifiers. To this end, we computed the gain-based feature importance. The gain-based feature importance directly measures the contribution of a feature to the reduction of the loss function. Table S4 in the Supplementary Material shows the three most important features for all classifiers.

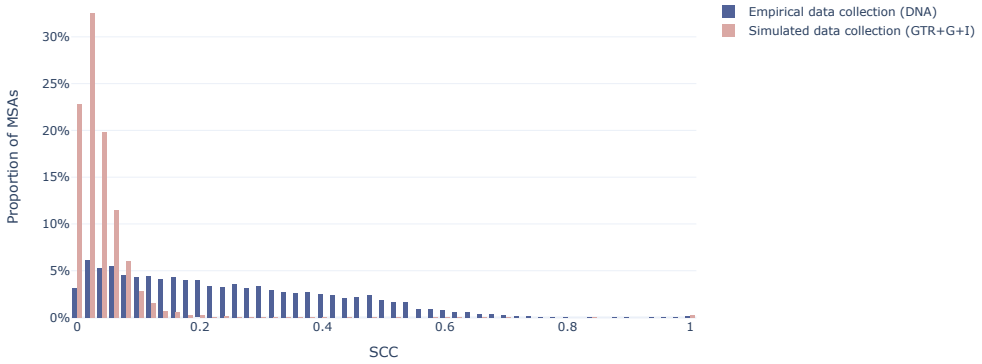
We observed that, except for one data collection, the *SCC* randomness metric was the most important feature. For classifying the LG+S256+GC+sparta data collection, it was the second most important feature. Figure 3 shows the distribution of SCC values for one example DNA data collection (GTR+G+I), as well as for one example protein data collection (LG+S256+GC) compared to the distribution for the respective empirical data collection. The lower the SCC value, the more random is the distribution of rates of evolution across sites in the MSA. The SCC values for simulated MSAs are substantially lower than for empirical MSAs. This shows that the rates of evolution across sites are more uniformly distributed in simulated MSAs compared to empirical MSAs, simulated data is thus more "random" than empirical data. We observed similar patterns for all other data collections as well.

We also frequently observed the *Entropy*, the *Pattern entropy*, as well as the *Bollback multinomial* metrics being among the three most important features. Analogously to the randomness features, these three features measure the randomness across sites of the MSA, albeit, not in an as fine-grained manner as the randomness features.

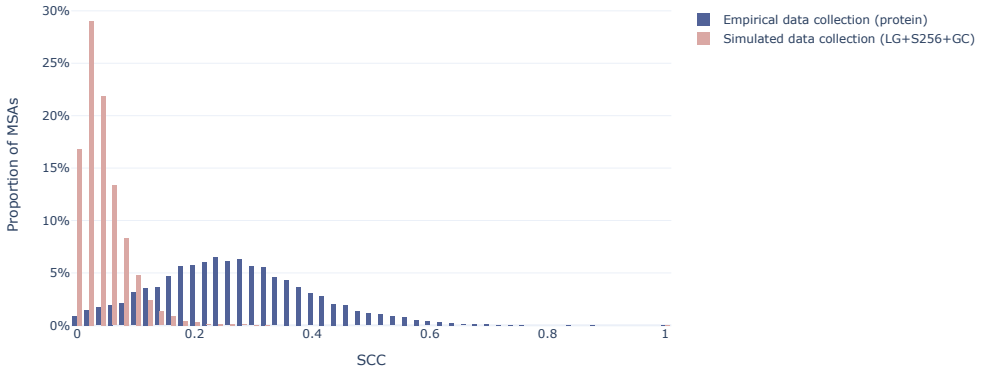
To gain further insights into the importance of the randomness features for classification, we additionally retrained all GBT classifiers without this set of randomness features. Table S6 in the Supplementary Material shows the resulting BACCs alongside the three most important features. As expected, the BACCs decrease for all data collections. Interestingly, the BACCs for the GTR+G and GTR+G+I DNA data collections decreased substantially from 0.89 to 0.65 and 0.61 respectively, yielding the prediction only marginally better than random guessing. Using this reduced set of features for the prediction, we observed interesting differences in feature distributions for features other than the randomness features. We observed that, compared to simulated data, empirical data tends to have a higher proportion of invariant sites (Figure 4a). The branch lengths in trees inferred for simulated MSAs tend to be shorter (Figure 4b; for better visualization, we only show data between the 10% and 90% percentile), and the *parsimony RF-Distance* tends to be higher for empirical data (Figure 4c). While Figure 4 depicts the distribution of feature values for one exemplary data collection (JC) only, these observations hold true for all simulated data collections. The more complex the model of evolution, the less pronounced these differences are, especially for the simulated DNA data under GTR+G and GTR+G+I (see Supplementary Material Figures S12 and S13).

We further explored the substantial decrease in accuracy for the GTR+G+I+mimick data with a BACC of 0.77. To this end, we split MSAs site-wise into 100 parts (buckets), averaged the number of substitutions per bucket (normalized by the maximum number of substitutions per MSA), and averaged the buckets over every MSA (see Figures S9 to S11 in the Supplementary Material). Interestingly, we could observe that the substitutions for empirical and the GTR+G+I+mimick data collections are concentrated at the beginning and the end of the MSAs, while the number of substitutions in GTR+G+I+sparta seem to be uniformly distributed. This also seemed to be the case for other substitution models (results not shown). This result is in agreement with Bricout et al. [10] who also found this pattern in a large scale analysis of empirical alignments.

As described above, we simulated the DNA data collections and the protein data collections without indels based on trees inferred using RAxML-NG. Trees from protein data with indels used for our indel simulations were inferred using IQ-Tree. For 10 out of 15 data collections, one of the branch length features were among the three most important features. To ensure that we did not leverage a tool induced bias for our prediction, we retrained all classifiers using only the MSA-based features by discarding all branch length features. We observed no substantial impact on the overall prediction accuracies. With GTR+G+I+mimick we observed the highest BACC difference. Using all features, the GBT achieved a prediction accuracy of 0.77. Discarding the branch length features resulted in a BACC of 0.74. Table S5 in the Supplementary Material shows the resulting BACCs for all classifiers, alongside the three most important prediction features when only using MSA based features.

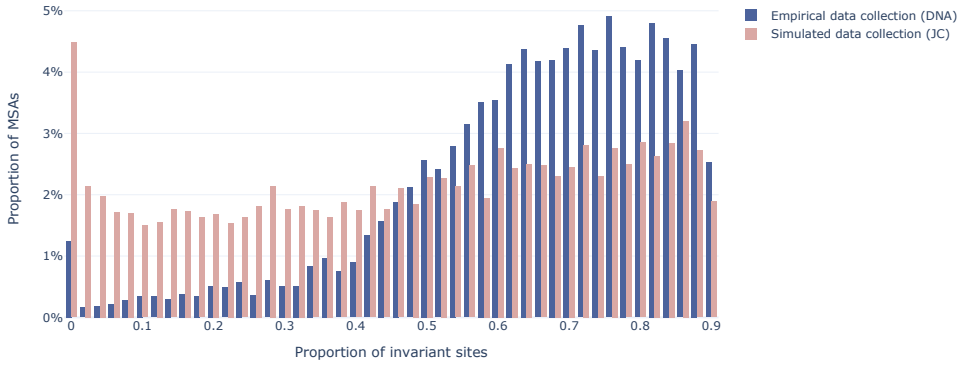


(a) Distribution of SCC values for the GTR+G+I and empirical DNA data collections.

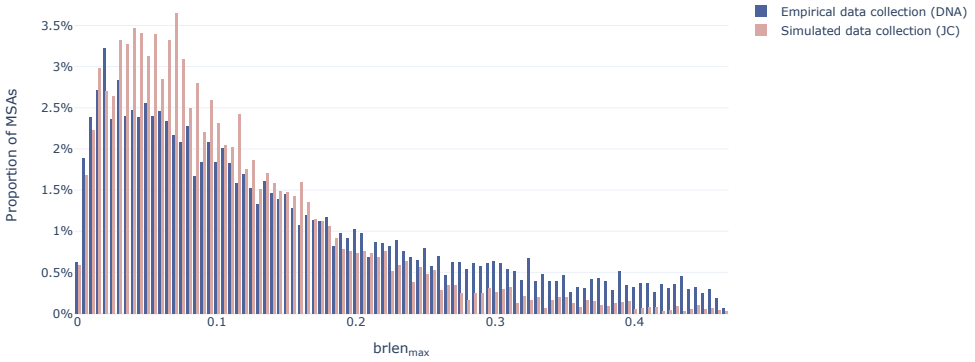


(b) Distribution of SCC feature values for the LG+S256+GC and protein empirical data collections.

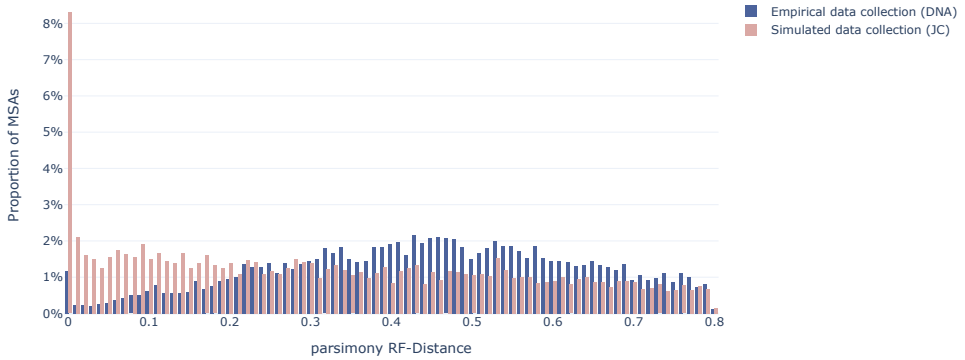
Figure 3: Feature distribution of SCC feature values for one exemplary DNA and Protein data collection. The dark blue bars represent the respective empirical data collection and the light pink bars represent the respective simulated data collection.



(a) Distribution of proportion of invariant feature values for the JC and empirical data collections.



(b) Distribution of $brlen_{max}$ feature values for the JC and empirical data collections.



(c) Distribution of *parsimony RF-Distance* feature values for the JC and empirical data collections.

Figure 4: Feature distribution for important features for classifying the JC data collection. The dark blue bars represent the empirical data collection and the light pink bars represent the simulated JC data collection.

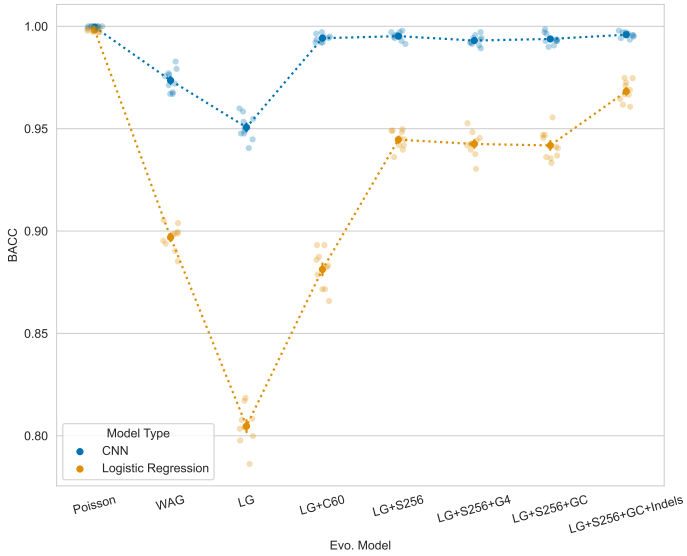


Figure 5: Performance of logistic regression on MSA compositions and CNN on site-wise compositions. For each evolutionary model the BACC of each fold is represented as well as the mean and standard error.

In addition to the feature analysis of the GBTs, we further investigated the remarkably accurate performance of the CNN on simulations using mixtures of stationary frequency profiles (i.e., the S256 or C60 model). Given that we could achieve better performance when using average global pooling, that is, averaging across the sequence, instead of maximum local pooling following the convolution layer (see paragraph CNN Architecture) we hypothesize that there must be predictive global features that aid in distinguishing simulated from empirical MSAs. To verify whether average alignment compositions constitutes a crucial factor, we trained logistic regression models to undertake the same classification task, but using site compositions averaged along the alignment i.e., MSA compositions. Figure 5 shows that the logistic regression model indeed performed well, particularly for simulated data under mixture models (BACC>0.94). Moreover, there is a strong correlation between BACCs of the CNNs and the logistic regression models ($r^2 = 0.85$).
We also attempted to train the logistic regression model on DNA data simulated under the GTR+G+I model, but found that there was no significant improvement during the first 100 epochs (BACC=0.51). Hence, the MSA composition is not sufficient for the classification of DNA data.

4 Discussion and Conclusion

In this study, we assessed the realism of sequence evolution simulations by attempting to discriminate between simulated MSAs and empirical MSAs using two distinct and independently developed classification methods. Specifically, we evaluated and interpreted the predictive accuracy of these approaches as a measure of realism. By addressing this question, we aimed to gain insights into the ability of current evolutionary models to accurately simulate evolutionary processes using continuous time Markov chains (CTMC).

Note that producing MSAs that are indistinguishable from empirical ones is a necessary but not sufficient condition for the degree of realism of a simulator. First, poor classification performance can occur because the classifier does simply not deploy appropriate functions or data representations. Hence, one can not guarantee that the simulated MSAs are realistic under all possible criteria. Second, poor performance can also be induced by optimization issues, especially when using deep learning methods. During our experiments, we observed low accuracies for CNNs several times. We managed to alleviate them by adapting the learning rate, the number of filters, or the pooling method, for instance. We thus advise researchers interested in classification performance as a realism metric to closely mon-

itor indicators of poor optimization, in particular, learning curves and gradient norms—in our case, poor optimization also led to a larger variance across folds and discrepancies in accuracy for the two classes. Because we found that all simulated MSAs were easy to discriminate from empirical MSAs, and because our results are consistent across two technically substantially distinct and independent classification methods, we conclude with confidence that the simulated MSAs generated in our study are not realistic.

It is worth noting here that we originally chose to develop a CNN for the classification task, as it is able to capture local dependencies among sites. With a kernel size greater than one, the network could potentially benefit from these dependencies for classification, as they are present in empirical MSAs yet cannot be replicated with standard simulators. However, we discovered that for protein data even with a kernel size of one combined with global average pooling (as an alternative to the commonly used local maximum pooling), the CNN still yields accurate performance. This type of network primarily focuses on capturing global features while overlooking local among-site dependencies. Consequently, these choices enabled us to thoroughly explore the limitations of current sequence evolution simulation approaches and different evolutionary models beyond their unrealistic assumption of independently evolving sites.

Our study has several strengths, such as the use of two fundamentally different classifiers, which allow for a broader assessment of possible weaknesses of current sequence evolution simulations: GBTs rely upon diverse, yet well-defined MSA properties, such as branch lengths or the randomness features that take into account the assumption of homogeneity along MSAs in classical simulations. The CNN only considers site-wise composition vectors, and thus exploits a signal that is not directly exploited by the GBTs. Given the high feature importance of the evolutionary rates (SCC) in the MSA, our GBTs unravel a lack of structure along simulated MSAs. The high classification accuracy of the CNNs, as well as our additional logistic regression analyses, show that simulated protein MSAs fail to reproduce global and site-specific alignment composition properties.

Furthermore, for the classification we used diverse and representative empirical protein and DNA databases: TreeBASE comprises representative data sets that are commonly analyzed in the field because it only contains MSAs of published studies, whereas HOGENOM offers a diverse sample of existing data, drawing from 499 nuclear Bacterial genomes, 46 from Archaea, and 121 from Eukaryotes.

Finally, we used phylogenetic trees reconstructed from these empirical data collections to simulate the data as realistically as possible. Thereby, we circumvented having to simulate realistic trees and can invoke simulations that are as similar as possible to the empirical MSAs. However, it is important to note that the realism of the simulations depends on the quality of the inferred phylogenetic trees when deploying this procedure. In particular, the inference of trees from protein MSAs without indels was performed under the assumption of homogeneous equilibrium frequencies (i.e., the LG substitution model). The resulting trees may have branches too short to be used for simulating MSAs in combination with site-heterogeneous mixture models. Tree inference under mixture models would typically yield longer branches than inference under the LG model. Consequently, this could result in unrealistically low evolutionary rates in simulations under mixture models. For the sake of simplicity, we treated the trees as given in this study. Nonetheless, we would like to emphasize that the combination of substitution models used to infer the trees and those employed to simulate the MSAs could be significant and warrants further investigation.

Overall, the classification task was not difficult, neither for DNA nor for protein data. Our CNN achieved an average BACC of 0.98 across all evolutionary models. This shows that existing models of sequence evolution fail to capture important characteristics of empirical site-wise compositions. In turn, this questions to which extent results obtained on simulated data apply to empirical data.

We originally hypothesized that with increasing evolutionary model complexity, classification performance would decrease. However, our results do not fully confirm this initial hypothesis. On the contrary, both classifiers remained highly accurate on the most complex evolutionary models for protein simulations with heterogeneous stationary distributions across sites. On DNA simulations, the inclusion of rate heterogeneity and a proportion of invariant sites did not lead to a substantial decrease in CNN classification accuracy either. Using the HKY substitution model instead of the JC model did also not result in more realistic simulations as a function of observed classification performance.

Finally, the most simple models, JC and Poisson, were classified with ease.

We used a state-of-the-art indel model with individual parameters for insertions and deletions and sampled indel parameters from approximated joint distributions. Nevertheless, both classifiers could again easily distinguish simulated from empirical data collections. In fact, classification accuracy substantially increased on DNA data *with* indels compared to data without indels (GTR+G+I). In contrast to this, using the *mimick* procedure to superimpose gaps onto simulated data appeared to result in more realistic MSAs. Yet, these MSAs could still be easily identified as simulated ones based on their site-wise compositions, as shown by the CNN results.

Furthermore, the prediction accuracy for protein data tended to be higher than the prediction accuracy for DNA data. We suspect that this is due to the higher number of states in the protein alphabet and therefore the increased number of possible patterns in a protein MSA, which yields a realistic simulation more challenging.

Our findings suggest that simulators based on classical site- and position-independent Continuous Time Markov Chains might not be able to generate data collections that appropriately resemble global low level site composition features of empirical DNA or protein data collections under existing evolutionary models. Considering the high importance of randomness related features for the GBT classifiers, and the respective feature value distributions, we conclude that the rate of evolution across sites of simulated MSAs are generated more uniformly along the MSA compared to empirical MSAs. For instance, simulations cannot reproduce the serial correlation of evolutionary rates that is present in empirical MSAs. We further observe that the proportion of invariant sites in classical simulations reduces their realism as measured by GBT. Apart from these findings, the CNN results reveal that there are unrealistic properties in terms of site-wise compositions that are independent of correlations among neighboring sites, which are not captured by the GBT classifier.

Moreover, the unexpectedly high accuracy of the CNN on simulations under mixture models that produce heterogeneous stationary distributions across sites indicates that these models produce global site composition features that the CNN can exploit to identify simulated MSAs. We suggest that the average MSA composition *is* one such global feature. However, these models are known to increase phylogenetic reconstruction accuracy, for instance, by partially alleviating long branch attraction artifacts [29]. The reasons for the suspected discrepancy between the average compositions of simulated MSAs and the empirical MSA compositions when empirical mixture models are used for simulations need to be further investigated.

If the reasons for this specific behavior of current simulators can be identified and eliminated, an adapted CNN architecture could be deployed to assess the importance of local site dependencies, as they are not included in standard simulators.

Simulated data that can not be distinguished from empirical data would be extremely useful for training, evaluating, and improving phylogenetic reconstruction methods [34, 28]. We conclude that a substantial amount of research remains to be conducted for improving substitution as well as indel evolution models, for both, protein and DNA data.

5 Acknowledgements

We thank Philippe Veber, research engineer at the Laboratoire de Biométrie et Biologie Évolutive, for providing us with the source code of his simulator. We gratefully acknowledge support from the CNRS/IN2P3 Computing Center (Lyon - France) for providing computing and data-processing resources needed for this work. This work was granted access to the HPC resources of IDRIS under the allocation 2022-AD011011137R2 made by GENCI. This work was financially supported by the Klaus Tschira Foundation, the ANR grant PIECES ANR-20-CE45-0017, and by the European Union (EU) under Grant Agreement No 101087081 (Comp-Biodiv-GR).

Funded by the European Union

References

[1] S. Abadi, O. Avram, S. Rosset, T. Pupko, and I. Mayrose. ModelTeller: Model selection for optimal phylogenetic reconstruction using machine learning. *Molecular Biology and Evolution*, 37 (11):3338–3352, 2020. doi: 10.1093/molbev/msaa154.

[2] H. Abdi. Bonferroni and šidák corrections for multiple comparisons. *Encyclopedia of measurement and statistics*, 3(01):2007, 2007.

[3] A. F. Agarap. Deep learning using rectified linear units (ReLU). *arXiv preprint arXiv:1803.08375*, 2018. doi: 10.48550/arXiv.1803.08375.

[4] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery. doi: 10.1145/3292500.3330701.

[5] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey. Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. *Nature biotechnology*, 33(8):831–838, 2015. doi: 10.1038/nbt.3300.

[6] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle. Deep learning for computational biology. *Molecular systems biology*, 12(7):878, 2016. doi: 10.15252/msb.20156651.

[7] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, page 2546–2554, Red Hook, NY, USA, 2011. Curran Associates Inc.

[8] J. P. Bollback. Bayesian model adequacy and choice in phylogenetics. *Molecular Biology and Evolution*, 19(7):1171–1180, 2002. doi: 10.1093/oxfordjournals.molbev.a004175.

[9] R. Bouckaert, T. G. Vaughan, J. Barido-Sottani, S. Duchêne, M. Fourment, A. Gavryushkina, J. Heled, G. Jones, D. Kühnert, N. De Maio, M. Matschiner, F. K. Mendes, N. F. Müller, H. A. Ogilvie, L. du Plessis, A. Popinga, A. Rambaut, D. Rasmussen, I. Siveroni, M. A. Suchard, C.-H. Wu, D. Xie, C. Zhang, T. Stadler, and A. J. Drummond. BEAST 2.5: An advanced software platform for bayesian evolutionary analysis. *PLOS Computational Biology*, 15(4):1–28, 2019. doi: 10.1371/journal.pcbi.1006650.

[10] R. Bricout, D. Weil, D. Stroebel, A. Genovesio, and H. R. Crollius. Evolution is not uniform along protein sequences. *bioRxiv*, pages 2022–04, 2022. doi: 10.1093/molbev/msad042.

[11] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann. The balanced accuracy and its posterior distribution. In *2010 20th International Conference on Pattern Recognition*, pages 3121–3124, 2010. doi: 10.1109/ICPR.2010.764.

[12] R. A. Cartwright. DNA assembly with gaps (Dawg): simulating sequence evolution. *Bioinformatics*, 21(Suppl 3):iii31–iii38, 2005. doi: 10.1093/bioinformatics/bti1200.

[13] K. Csilléry, M. G. Blum, O. E. Gaggiotti, and O. François. Approximate bayesian computation (ABC) in practice. *Trends in Ecology & Evolution*, 25(7):410–418, 2010. doi: 10.1016/j.tree.2010.04.001.

[14] J. S. Farris. Methods for computing wagner trees. *Systematic Biology*, 19(1):83–92, 1970. doi: 10.1093/sysbio/19.1.83.

[15] W. M. Fitch. Toward defining the course of evolution: Minimum change for a specific tree topology. *Systematic Zoology*, 20(4):406–416, 1971. doi: 10.2307/2412116.

[16] W. Fletcher and Z. Yang. INDELible: A flexible simulator of biological sequence evolution. *Molecular Biology and Evolution*, 26(8):1879–1888, 2009. doi: 10.1093/molbev/msp098.

[17] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232, 2001. doi: 10.1214/aos/1013203451.

[18] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[19] S. Guindon, J.-F. Dufayard, V. Lefort, M. Anisimova, W. Hordijk, and O. Gascuel. New algorithms and methods to estimate maximum-likelihood phylogenies: Assessing the performance of PhyML 3.0. *Systematic Biology*, 59(3):307–321, 2010. doi: 10.1093/sysbio/syq010.

[20] J. Haag, D. Höhler, B. Bettisworth, and A. Stamatakis. From easy to hopeless - predicting the difficulty of phylogenetic analyses. *bioRxiv*, 2022. doi: 10.1101/2022.06.20.496790.

[21] M. Hasegawa, H. Kishino, and T.-a. Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 22:160–174, 1985. doi: 10.1007/BF02101694.

[22] D. Hoehler, J. Haag, A. M. Kozlov, and A. Stamatakis. A representative performance assessment of maximum likelihood based phylogenetic inference tools. *bioRxiv*, 2022. doi: 10.1101/2022.10.31.514545.

[23] S. Höhna, M. J. Landis, T. A. Heath, B. Boussau, N. Lartillot, B. R. Moore, J. P. Huelsenbeck, and F. Ronquist. RevBayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification language. *Systematic Biology*, 65(4):726–736, 2016. doi: 10.1093/sysbio/syw021.

[24] T. H. Jukes and C. R. Cantor. Chapter 24 - evolution of protein molecules. In H. Munro, editor, *Mammalian Protein Metabolism*, pages 21–132. Academic Press, 1969. doi: 10.1016/B978-1-4832-3211-9.50009-7.

[25] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. doi: 10.48550/arXiv.1412.6980. URL http://arxiv.org/abs/1412.6980.

[27] D. Knuth. *Art of Computer Programming, The: Seminumerical Algorithms, Volume 2*. Addison-Wesley Professional, hardcover edition, 1997.

[28] A. M. Kozlov, D. Darriba, T. Flouri, B. Morel, and A. Stamatakis. RAxML-NG: a fast, scalable and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics*, 35(21):4453–4455, 2019. doi: 10.1093/bioinformatics/btz305.

[29] N. Lartillot and H. Philippe. A bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. *Molecular biology and evolution*, 21(6):1095–1109, 2004. doi: 10.1093/molbev/msh112.

[30] S. Q. Le and O. Gascuel. An improved general amino acid replacement matrix. *Molecular biology and evolution*, 25(7):1307–1320, 2008.

[31] G. Loewenthal, D. Rapoport, O. Avram, A. Moshe, E. Wygoda, A. Itzkovitch, O. Israeli, D. Azouri, R. A. Cartwright, I. Mayrose, et al. A probabilistic model for indel evolution: differentiating insertions from deletions. *Molecular biology and evolution*, 38(12):5769–5781, 2021. doi: 10.1093/molbev/msab266.

[32] N. Ly-Trong, S. Naser-Khdour, R. Lanfear, and B. Q. Minh. Alisim: A fast and versatile phylogenetic sequence simulator for the genomic era. *Molecular Biology and Evolution*, 39(5):msac092, 2022.

[33] B. Q. Minh, H. A. Schmidt, O. Chernomor, D. Schrempf, M. D. Woodhams, A. von Haeseler, and R. Lanfear. IQ-TREE 2: New models and efficient methods for phylogenetic inference in the genomic era. *Molecular Biology and Evolution*, 37(5):1530–1534, 2020. doi: 10.1093/molbev/msaa015.

[34] L. Nesterenko, B. Boussau, and L. Jacob. Phyloformer: towards fast and accurate phylogeny estimation with self-attention networks. *bioRxiv*, 2022. doi: 10.1101/2022.06.24.496975.

[35] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS 2017 Workshop on Autodiff*, 2017. URL https://openreview.net/forum?id=BJJsrmfCZ.

[36] S. Penel, A.-M. Arigon, J.-F. Dufayard, A.-S. Sertier, V. Daubin, L. Duret, M. Gouy, and G. Perrière. Databases of homologous gene families for comparative genomics. In *BMC bioinformatics*, volume 10(Suppl 6), pages 1–13. BioMed Central, 2009.

[37] W. H. Piel, L. Chan, M. J. Dominus, J. Ruan, R. A. Vos, and V. Tannen. TreeBASE v.2: A database of phylogenetic knowledge. *e-BioSphere 2009*, 2009.

[38] L. Prechelt. Early stopping—but when? *Neural networks: tricks of the trade: second edition*, pages 53–67, 2012. doi: 10.1007/978-3-642-35289-8_5.

[39] M. N. Price, P. S. Dehal, and A. P. Arkin. Fasttree: Computing large minimum evolution trees with profiles instead of a distance matrix. *Molecular Biology and Evolution*, 26(7):1641–1650, 2009. doi: 10.1093/molbev/msp077.

[40] D. Robinson and L. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1): 131–147, 1981. doi: 10.1016/0025-5564(81)90043-2.

[41] D. Schrempf, N. Lartillot, and G. Szöllősi. Scalable empirical mixture models that account for across-site compositional heterogeneity. *Molecular Biology and Evolution*, 37(12):3616–3631, 2020. doi: 10.1093/molbev/msaa145.

[42] D. W. Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015. doi: 10.1002/9780470316849.

[43] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

[44] J. Shoemaker and W. Fitch. Evidence from nuclear sequences that invariable sites should be considered when sequence divergence is calculated. *Molecular Biology and Evolution*, 6(3):270–289, 1989. doi: 10.1093/oxfordjournals.molbev.a040550.

[45] L. Si Quang, O. Gascuel, and N. Lartillot. Empirical profile mixture models for phylogenetic reconstruction. *Bioinformatics*, 24(20):2317–2323, 2008. doi: 10.1093/bioinformatics/btn445.

[46] L. N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017. doi: 10.1109/WACV.2017.58.

[47] S. Tavaré. Some probabilistic and statistical problems on the analysis of DNA sequences. *Lectures on Mathematics in the Life Sciences*, 17:57–86, 1986.

[48] G. Van Rossum and F. L. Drake Jr. *Python tutorial*, volume 620. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

[49] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[50] R. A. Vos, J. P. Balhoff, J. A. Caravas, M. T. Holder, H. Lapp, W. P. Maddison, P. E. Midford, A. Priyam, J. Sukumaran, X. Xia, et al. Nexml: rich, extensible, and verifiable representation of comparative data and metadata. *Systematic biology*, 61(4):675–689, 2012. doi: 10.1093/sysbio/sys025.

[51] S. Whelan and N. Goldman. A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach. *Molecular biology and evolution*, 18(5): 691–699, 2001. doi: 10.1093/oxfordjournals.molbev.a003851.

[52] Z. Yang. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *Journal of Molecular evolution*, 39:306–314, 1994. doi: 10.1007/BF00160154.

[53] J. Zhou and O. G. Troyanskaya. Predicting effects of noncoding variants with deep learning–based sequence model. *Nature methods*, 12(10):931–934, 2015. doi: 10.1038/nmeth.3547.