

Novel Protein Evolution Models for Ancestral Sequence Reconstruction

Akshay Ravor



Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-190

<http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-190.html>

September 23, 2024

Copyright © 2024, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Acknowledgement

I would like to thank Yun S. Song and Sebastian Prillo for introducing me to computational biology at Berkeley and for their guidance throughout the course of the project. Sebastian also provided much of the initial code for the project and the SiteRM model. I am also grateful to Antoine Koehl for his help on the deep learning aspects of the project and the RNN model, as well as to Matthew Liu for helping me obtain the viral datasets I considered for evaluation. Finally, thank you to my parents for supporting me during this long journey.

Novel Protein Evolution Models for Ancestral Sequence Reconstruction

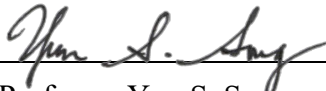
by Akshay Ravoor

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences,
University of California at Berkeley, in partial satisfaction of the requirements for the
degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:



Professor Yun S. Song
Research Advisor

September 19, 2024

(Date)



Professor John P. Huelsenbeck
Second Reader

September 23, 2024

(Date)

Novel Protein Evolution Models for Ancestral Sequence Reconstruction

Copyright 2024
by
Akshay Ravoor

Abstract

Novel Protein Evolution Models for Ancestral Sequence Reconstruction

by

Akshay Ravoor

Master of Science in Electrical Engineering & Computer Science

University of California, Berkeley

Professor Yun S. Song, Chair

Models of protein evolution are essential for a variety of applications, from phylogenetic analysis and ancestral sequence reconstruction to variant effect prediction and protein design. A protein evolution model is typically characterized by a rate matrix Q which describes the rate at which amino acids mutate into one another, and evolution is inferred under an independent sites model (possibly with site rate variation). In this work we move beyond classical models by leveraging the CherryML framework to efficiently train new kinds of models either doing away with the independent sites assumption (RNN) or the idea of global rate matrices (SiteRM). Both the RNN and SiteRM models show improved performance compared to WAG when evaluated on per-site likelihood. We then apply these two models to ASR using the extant sequence reconstruction method and a variety of reconstruction algorithms. We are able to use the SiteRM model to attain a performance competitive with IQ-Tree and consistently outperform it in the longer sequence length datasets. Though more validation is needed for the particular task of ASR, our results are promising for the development of new protein evolution models under the CherryML paradigm.

Contents

| | |
|---|------------|
| Contents | i |
| List of Figures | ii |
| List of Tables | iii |
| 1 Introduction | 1 |
| 1.1 Protein Evolution Models | 1 |
| 1.2 Ancestral Sequence Reconstruction | 3 |
| 1.3 Problem Statement | 6 |
| 2 Novel Models of Protein Evolution | 7 |
| 2.1 CherryML | 7 |
| 2.2 Training and Testing Data | 8 |
| 2.3 RNN Model | 9 |
| 2.4 SiteRM Model | 10 |
| 2.5 Results | 10 |
| 3 Ancestral Sequence Reconstruction | 12 |
| 3.1 Datasets | 12 |
| 3.2 Methods | 15 |
| 3.3 Results | 17 |
| 4 Conclusion | 20 |
| 4.1 Discussion | 20 |
| 4.2 Future Work | 20 |
| Bibliography | 22 |

List of Figures

| | | |
|-----|---|----|
| 1.1 | Illustration of Ancestral Sequence Reconstruction | 4 |
| 2.1 | Illustration of CherryML Methodology | 8 |
| 2.2 | Comparison of Train/Test Transition Times | 9 |
| 2.3 | Model Performance on In-Family and Out-Family Sequences | 11 |
| 3.1 | Illustration of Extant Sequence Reconstruction | 14 |
| 3.2 | Performance on ASR | 19 |

List of Tables

| | | |
|-----|--|----|
| 3.1 | Summary Statistics of ASR Datasets | 15 |
|-----|--|----|

Acknowledgments

I would like to thank Yun S. Song and Sebastian Prillo for introducing me to computational biology at Berkeley and for their guidance throughout the course of the project. Sebastian also provided much of the initial code for the project and the SiteRM model. I am also grateful to Antoine Koehl for his help on the deep learning aspects of the project and the RNN model, as well as to Matthew Liu for helping me obtain the viral datasets I considered for evaluation. Finally, thank you to my parents for supporting me during this long journey.

Chapter 1

Introduction

1.1 Protein Evolution Models

Understanding the evolution of proteins over time is crucial to a variety of applications in biology such as phylogenetic tree reconstruction, ancestral sequence reconstruction, multiple sequence alignments, variant effect prediction, and protein design. Such protein evolution models can differ in their level of molecular granularity (DNA, RNA, amino acid, or codons), use of site rate variation (invariable sites, Γ distributed site rates, or the probability-distribution-free model), the treatment of insertions and deletions, and assumptions of i.i.d (independent and identically distributed) sites.

Classically, the evolution of amino acids is described by phylogenetic models parametrized by a 20 by 20 rate matrix Q under the assumption that molecules evolve down a phylogenetic tree according to a continuous-time Markov process. The rows and columns of Q are indexed by the 20 canonical amino acids present in most living beings, and the matrix describes the rate at which each amino acid mutates into another. For example, given such a rate matrix Q we can estimate the probability of the amino acid alanine (A) mutating into leucine (L) after time t :

$$p_Q(\text{A} \xrightarrow{t} \text{L}) = \exp(Qt)_{i,j}$$

where the i th row corresponds to alanine and the j th column corresponds to leucine. Classical models make the further assumption of i.i.d sites, so the likelihood of protein sequence y of length L evolving from sequence x also of length L after time t under the above rate matrix Q would be:

$$p(y|x, t) = \prod_{i=1}^L p(y_i|x_i, t) = \prod_{i=1}^L \exp(Qt)_{x_i, y_i}$$

Formally, a model of protein evolution is a conditional distribution $p(y|x, t, c)$ that describes the probability of sequence y arising from sequence x after time t with context c

(which could be the protein family f , information about the structure of the protein, etc.). As we will see models need not all prescribe to the above approach of using the same rate matrix for each site. Indeed, a model of protein evolution need not use rate matrices at all.

Early Models

Early approaches to estimating the rate matrix for protein evolution relied on counting methods. The Dayhoff model [4] first generates a series of phylogenetic trees over multiple MSAs (multiple sequence alignments). It then uses maximum parsimony to assign sequences to ancestral states. Finally, it simply counts the number of transitions between each pair of amino acids across all branches of the tree to produce a rate matrix.

A later model by Jones, Taylor, and Thornton (the JTT model) [13] estimates a rate matrix with the same counting approach, but instead of using maximum parsimony to infer ancestral states it uses an iterative process to count mutations and then prune pairs of nearest neighbors in the estimated tree.

Both methods are biased as they assume a single mutation per site per branch and do not account for evolutionary time (branch lengths) because of their reliance on a counting based approach to matrix estimation.

Probabilistic Models

The first probabilistic (as opposed to counting based) model of protein evolution was proposed by Whelan and Goldman (WAG) [31]. WAG takes a coordinate ascent approach by alternating between estimating the maximum likelihood phylogenetic tree under the rate matrix and estimating the maximum likelihood rate matrix from the tree. The model assumes that each site evolves i.i.d. following a time-reversible continuous-time Markov Chain parametrized by a global 20 by 20 transition rate matrix Q (which it seeks to estimate). The resulting optimization problem takes the following form:

$$\arg \max_{Q,T} P(D|T, Q) = \arg \max_{Q,T} \prod_{i=1}^m P(D_i|T_i, Q)$$

where $D = (D_1, D_2, \dots, D_m)$ are the m MSAs and $T = (T_1, T_2, \dots, T_m)$ are the estimated trees for each MSA. However, finding the MLE phylogenetic tree for an MSA can be very computationally expensive. WAG therefore uses a neighbor-joining tree estimation method as a proxy for maximizing the likelihood before performing a zeroth-order optimization on the branch lengths. Thus, given the estimated WAG matrix Q one can calculate the transition likelihood of the model:

$$p_{\text{WAG}}(y|x, t, f) = \prod_{i=1}^{L_f} \exp(\alpha_i^f Q t)_{x_i, y_i}$$

where the α 's can be used to account for differing rates of evolution for each protein family f and L_f is the alignment length of the MSA for protein family f . This methodology enabled the WAG method to estimate rate matrices for much orders of magnitude more sequences across more MSAs than previous MLE-based methods.

The seminal Le-Gascuel (LG) model [16] builds upon WAG by doing away with the constant sites assumption and incorporates site rate variation by positing that sites in a protein evolve under scalar multiples of the global transition-rate matrix Q . The potentially differing scalars ($\alpha_1^f, \alpha_2^f, \dots, \alpha_{L_f}^f$) for each site are known as the site rates. Thus the likelihood under the LG model is:

$$p_{\text{LG}}(y|x, t, f) = \prod_{i=1}^{L_f} \exp(\alpha_i^f Q t)_{x_i, y_i}$$

In order to limit the number of model parameters ($400 + \sum_{f=1}^m L_f$ where m denotes the total number of protein families; notation from [22]), the LG model will use a pre-defined number of rate categories (e.g. 20) indicating that each site rate α_i^f will belong to one of these categories (so the model only has to estimate a number of site rates equal to the number of rate categories). Furthermore, the rate categories are often constrained to follow some distribution such as the Γ model of site-rate variation.

Nonetheless, the LG model remains one of the most popular models of protein evolution for phylogenetic tree reconstruction and is heavily utilized in popular libraries like IQ-Tree [17] and PhyML [9].

1.2 Ancestral Sequence Reconstruction

A primary motivation behind the development of molecular evolution models is to understand how genes and proteins evolve in response to changing environments. Without intact DNA evidence for true ancestral sequences (e.g. from fossils), scientists rely on the development of such protein evolution models to infer the ancestral sequences of extant proteins. Ancestral sequence reconstruction (ASR) has not only been used to draw insights into the early history of life on Earth, but has more recently been leveraged as a tool for designing novel proteins. Ancestral sequences often possess several valuable properties like thermostability and broad substrate binding range while also preserving the desirable properties of their extant descendants [10]. Such approaches have greatly benefited from advances in DNA sequencing technology and computational phylogenetic analysis and have been put to use in molecular ecology, the development of vaccines for viral diseases, and the discovery of novel substances in biotechnology and pharmaceutical companies [24, 3, 26, 2, 6, 35, 34, 12].

Figure 1.1 shows the most common formulation of an ASR problem. A set of known extant sequences shown as red squares labelled with letters are the leaves of a phylogenetic tree. We seek to infer (reconstruct) the internal nodes (ancestors) of the tree which are shown as yellow circles labelled with numbers. Time flows down the tree from the root node

(oldest ancestral sequence) to the leaves (extant sequences), with branch lengths usually corresponding to some measurement of evolutionary time. Note that the branch lengths are not shown to scale; the extant sequences need not have arisen at exactly the same time.

In many cases, datasets consist of only extant sequences and it falls on researchers to infer the tree topology. Thus ASR methods will often have to search over the space of phylogenetic trees before reconstructing ancestral nodes. Sometimes only the reconstructed sequence of the root node (the “oldest” ancestor) is desired, while other times the trajectory of evolution along the tree is of interest as in the study of viral disease.

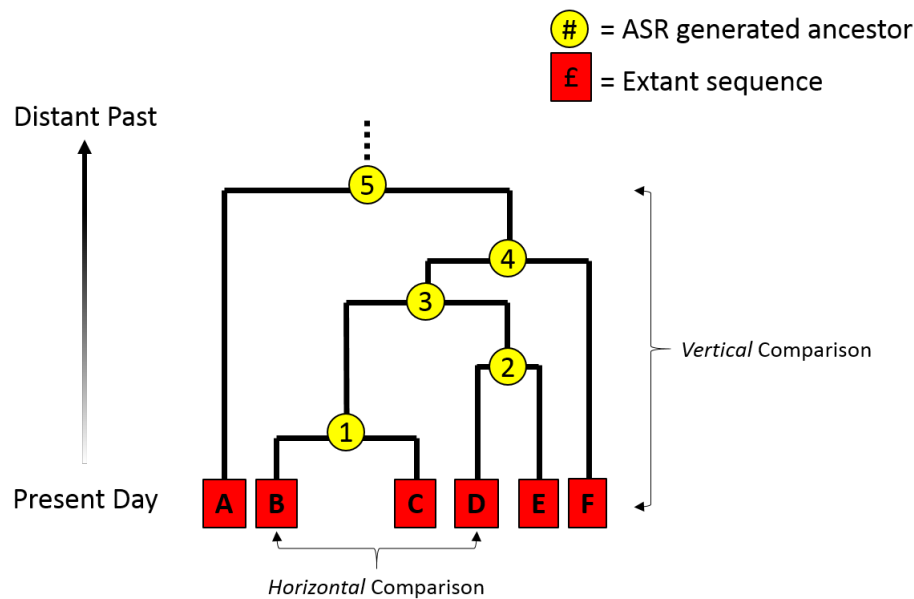


Figure 1.1: An example of a phylogenetic tree associated with the ASR problem. Yellow nodes are inferred from the tree topology and known sequences of the red nodes. Branch lengths are not to scale. Image taken from [5].

In this work we delegate the reconstruction of the tree to other programs and focus just on the process of reconstructing ancestral sequences under a fixed topology.

Maximum Parsimony

Among the first and conceptually simple reconstruction methods was maximum parsimony, with the earliest implementation being Fitch’s algorithm [8].

Parsimony methods seek to assign ancestral character states in a given tree so as to minimize the total number of character state changes necessary to produce the states observed at the leaves. Scientists can also perform weighted parsimony which uses some cost function (e.g. a rate matrix) to penalize character state changes instead of all transitions being equally costly as in unweighted parsimony.

Although maximum parsimony methods are simple, efficient, and sometimes comparable in accuracy to other more complex reconstruction algorithms (described in subsequent sections), its use is presently relatively limited [28]. Parsimony ignores branch lengths and so does not account for variation in time among lineages. Parsimony also assumes the minimum amount of character changes imposed by the data meaning the amount of homoplasy is minimized. This may not be appropriate under some rapid evolution settings. Lastly, parsimony methods have no statistical justification and are not statistically consistent.

Maximum Likelihood ASR

Following prior parsimony based methods, maximum likelihood (ML) became the go-to method for reconstructing protein ancestral states in a phylogeny [7, 33, 15]. By leveraging a probabilistic model of protein evolution, ML allowed for a more accurate characterization of ancient sequences.

Like parsimony, ML requires a phylogenetic tree and the extant sequences at the leaves of the tree. Unlike parsimony, ML also requires a model of protein evolution, the simplest of which is a single substitution matrix like in the WAG model [31]. Consequently the tree must also have resolved branch lengths. Indeed, this can be seen as a strength of ML since parsimony methods completely ignore branch lengths and so do not account for the evolutionary distance between two nodes. Furthermore, the use of an explicit protein evolution model can account for the fact that not all transitions are equally likely (e.g. positively charged amino acids are more likely to mutate into another positively charged amino acids than a negatively charged one).

Marginal methods reconstruct the sequence for a single node using marginal likelihoods that integrate over the probabilities of amino acids in other nodes of the tree. Consider a binary phylogenetic tree with some node of interest u (perhaps the root) and two descendants u_1 and u_2 with respective branch lengths t_1 and t_2 . First let us consider reconstruction of a single ancestral character with $L_u(i)$ as the conditional probability of observing the data at the leaves of the subtree rooted at node u given that the state of u is i . Then with some basic protein evolution model $p(y|x, t)$ and Ω as the set of available character states we can write:

$$L_u(i) = \left(\sum_{j_1 \in \Omega} p(j_1|i, t_1) \cdot L_{u_1}(j_1) \right) \left(\sum_{j_2 \in \Omega} p(j_2|i, t_2) \cdot L_{u_2}(j_2) \right)$$

which is known as Felsenstein's pruning algorithm [7].

The probabilistic approach considers two types of ancestral reconstruction: the *marginal* and *joint* reconstructions. The above is the marginal reconstruction and is akin to a greedy optimization since we locally optimize the likelihoods of each ancestral node but do not necessarily arrive at the globally optimal solution to the problem.

One can also incorporate a prior to arrive at the probabilities $P(u = i|S)$ of each state i of a given ancestral node u , where S represents the observed values at all leaves of the tree. Using Bayes formula, we obtain:

$$p(u = i|S) = \frac{p(u = i)p(S|u = i)}{P(S)} = \frac{\pi_i p(S|u = i)}{P(S)}$$

where π_i is the equilibrium frequency of character i . The likelihood of the data can be obtained from the above steps with Felsenstein's pruning algorithm [7].

A subsequent methodological development brought about a joint reconstruction method across all nodes, thereby maximizing the likelihood of the entire dataset [23]. While joint reconstruction conceptually provides a maximum likelihood method for providing a complete evolutionary history of each site, it is not widely used in practice [28]. This is in part because empirical research indicates that marginal reconstruction is more suitable when one wants the sequence at a particular node, like in molecular restoration studies, whereas joint reconstruction is more suitable when one counts changes at each site [25]. Especially for molecular recovery tasks, marginal reconstruction is the preferred method [25].

1.3 Problem Statement

In this work, we pursue two primary goals:

1. In chapter 2, we seek to develop novel models of protein evolution under the framework of $p(y|x, t, c)$, doing away with the assumptions of prior works that a family must be parameterized under a single rate matrix with various scaling factors (rate categories). Furthermore, we attempt to move away from the independent-sites model of evolution.
2. In chapter 3, we test these new models on an important downstream task, ancestral sequence reconstruction (ASR). We compare their performance with existing ASR tools.

Chapter 2

Novel Models of Protein Evolution

2.1 CherryML

While the LG estimation method greatly improved the speed and accuracy of estimating MLE rate matrices, it is still bottlenecked by the process of alternately estimating the entire tree topology and rate matrices due to the need to marginalize ancestral states of each tree (which is usually accomplished with Felsenstein’s pruning algorithm [7]). To estimate our protein evolution models we leverage a recent method called CherryML [21] which replaces the full joint likelihood over the tree with a composite likelihood over just the cherries. In practice, it turns out that the optimizing the maximum composite likelihood over cherries works just as well as the full likelihood, with a ~50% relative loss of statistical efficiency for the LG model while being over a thousand times faster [21]:

$$\arg \max_{Q,T} P(D|T, Q) \approx \arg \max_{Q,C} P(D|C, Q) = \arg \max_{Q,C} \prod_{(x,y) \in C} P_Q(x|y, t_{x \leftrightarrow y})$$

where C are the set of cherries in the tree T and we use a model of protein evolution Q which in this example is a rate matrix.

CherryML also quantizes transition times into buckets; the original paper states that a few hundred points are sufficient to achieve an error as low as 1% [21]. These two techniques—maximum composite likelihood over cherries and time quantization—are depicted in figure 2.1 and vastly speed up the process of estimating rate matrices for classical models. This method has been shown to improve the estimation of LG matrices by several orders of magnitude [21].

In this work, we leverage the results from CherryML to estimate new kinds of protein evolution models using the (x, y, t) triples drawn from cherries of the estimated tree as “datapoints”.

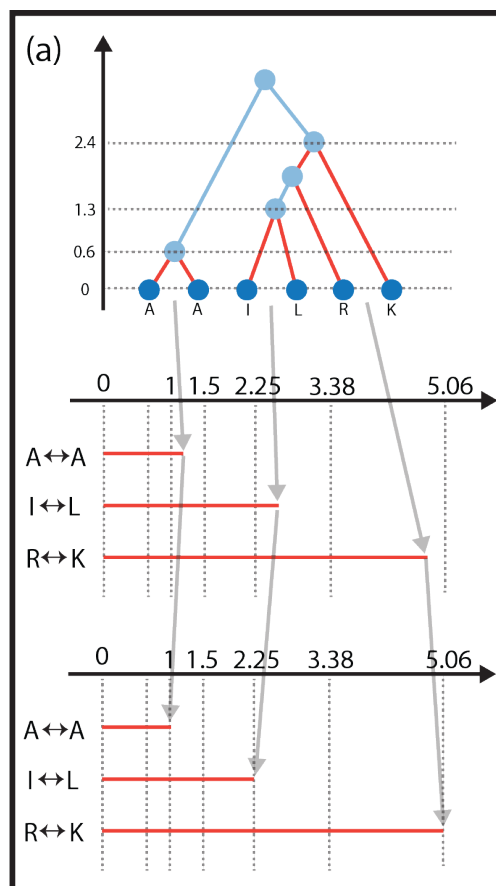


Figure 2.1: Image taken from CherryML paper [21]. Cherries (dark blue, joined by red branches) are iteratively pruned from the estimated tree and quantized into (x, y, t) transitions.

2.2 Training and Testing Data

We use protein families and sequences from the TrRosetta dataset [32] with each being sampled down to 1,024 sequences to speed up tree inference. Sequences consist of the 20 amino acids as well as a gap character ‘-’.

Trees are estimated using FastTree 2.1 [20] under the WAG model and a single rate category. Once estimated, each tree is split along a single edge to create two subtrees of roughly equal leaves. One tree is designated as the train tree, and its (x, y, t) transitions are in the train set for that family. The other tree is the test tree, and its transitions are in the test set for that family. Splitting the tree along a single edge instead of randomly assigning transitions to either the train or test set allows us to maximize the evolutionary divergence between the train and test sets while still retaining a similar distribution of transition times as shown in figure 2.2. In this way we can train a model on particular families while still

valuating it on held-out transitions for that same family.

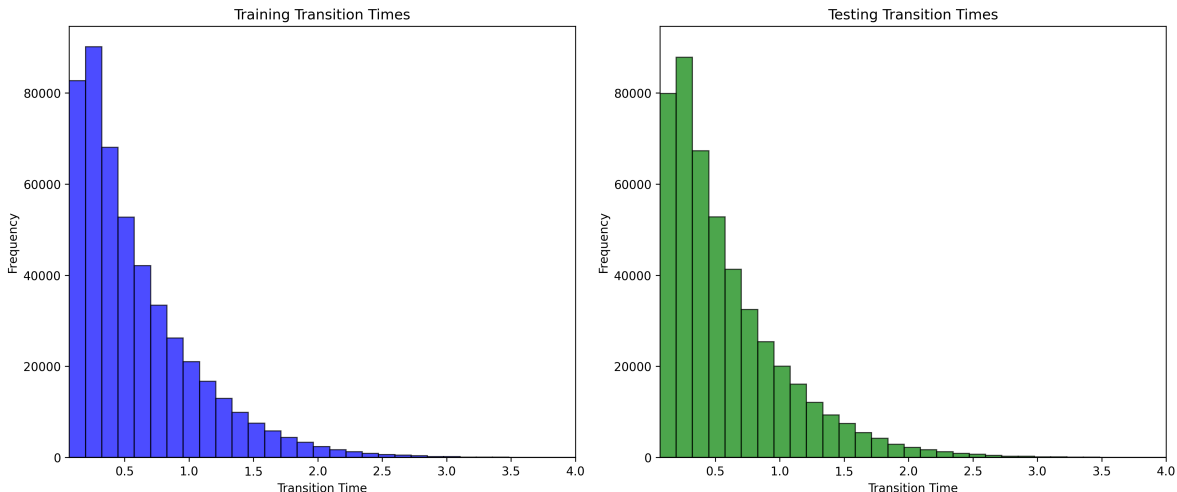


Figure 2.2: Comparison of transition times between the test and train dataset as drawn from 1000 random families.

2.3 RNN Model

To stay relatively close to proven and tested classical models, we use an RNN decoder that attempts to strictly generalize the WAG model [14]. The RNN uses the context of previously predicted positions in a sequence to predict a rate matrix at each subsequent position, which then goes through a matrix exponential layer to give a distribution over output tokens based on the input sequence at that position in x . More formally, with h_i being the hidden state of the RNN at position i , we have:

$$p(y_i|y[:i-1], x[:i], t) = \exp(tQ(h_i))_{x_i, y_i}$$

where $Q(h_i)$ is a contextual site-specific rate matrix outputted by the RNN to decode position i . It is worth noting that setting the recurrent connections h_i to 0 the RNN recovers exactly the WAG model. In this way we hope that the RNN can learn updates to the WAG model instead of vastly overfit site-specific rate matrices. This also influenced the decision to not use x_i as input to the generation of the rate matrix as position i .

To get around having to optimize the rate matrix directly (a constrained optimization problem), we instead parameterize the rate matrix Q with a vector θ and an upper triangular matrix Θ as described in CherryML [21].

Letting $\pi = \text{Softmax}(\theta)$ and $S = \text{Softplus}(\Theta + \Theta^T)$ we take the off-diagonal entries of Q to be $\sqrt{\frac{1}{\pi}} S \sqrt{\pi}$ (where operations are performed entry-wise). The diagonal entries of Q

are then uniquely determined. Thus the neural network is actually adding an unconstrained error term to θ and Θ instead of the final rate matrix Q . We can then directly optimize θ and Θ in an unconstrained fashion.

Our best performing architecture was a 2-layer LSTM with a hidden state size of 1024. We use 1,000 randomly selected families in our training set for the RNN model.

2.4 SiteRM Model

We also use a protein evolution model that learns a distinct rate matrix for each site of a specific protein family, hereafter referred to as the SiteRM model [22]. With Q_1^f, \dots, Q_L^f as site-specific rate matrices for an MSA of length L from family f , our model predicts:

$$p_{\text{SiteRM}}(y|x, t, f) = \prod_{i=1}^L \exp(tQ_i^f)_{x_i, y_i}$$

Given the training transitions for a family, we use CherryML’s transition-rate matrix estimation procedure separately at each site to get the site-specific rate matrices Q_i^f . Since the model is heavily over-parameterized due to there being only ≈ 256 pairs of amino acids at each site in a family (1024 sequences, 512 in the train set, 256 leaves), we regularize the site-specific rate matrices with the LG model rate matrix under 20 rate categories. We find that in practice a mixing coefficient of $\lambda = 0.5$ works quite well. Using CherryML, we are able to estimate a rate matrix for a site in a family approximately every second.

We trained the SiteRM model on the same 1,000 families used to train the RNN. By nature of the model, it does not generalize at all to unseen families.

2.5 Results

To evaluate the performance of our new models we aggregate per-site transition likelihoods over the 1,000 training families (in-families) and another 1,000 held-out families (families for which the models have never seen either the train or test set).

It has been shown that large contiguous sequences of gap characters resulting from insertion and deletion events can trivialize prediction for an autoregressive model like the RNN (it can mindlessly keep predicting gaps) [14]. To get around this, we renormalize the likelihood conditioned on the true character at that site not being a gap character [14]. In this way we eliminate the influence of models “cheating” by predicting gaps when in a gappy region and assess only their performance in non-gap regions.

Figure 2.3 shows the performance of the RNN and SiteRM model compared to the classical WAG model (all trained on the same 1,000 families). We observe that the RNN and SiteRM models exhibit strong performance on in-family transitions, both significantly outperforming WAG across all time bins. Unfortunately, the RNN performance drops to that of WAG on the held-out family transitions. Nevertheless, this does confirm that the RNN

model functions as a strict generalization of WAG: it outperforms WAG on families in its training set, but does no worse than WAG when presented with sequences from an unknown family.

Given that the RNN's performance boost is limited to families it has seen, we also evaluated a much smaller RNN that could be rapidly trained on a set of families of interest. However, this RNN exhibited decreased performance even on in-family transitions and the idea was discarded.

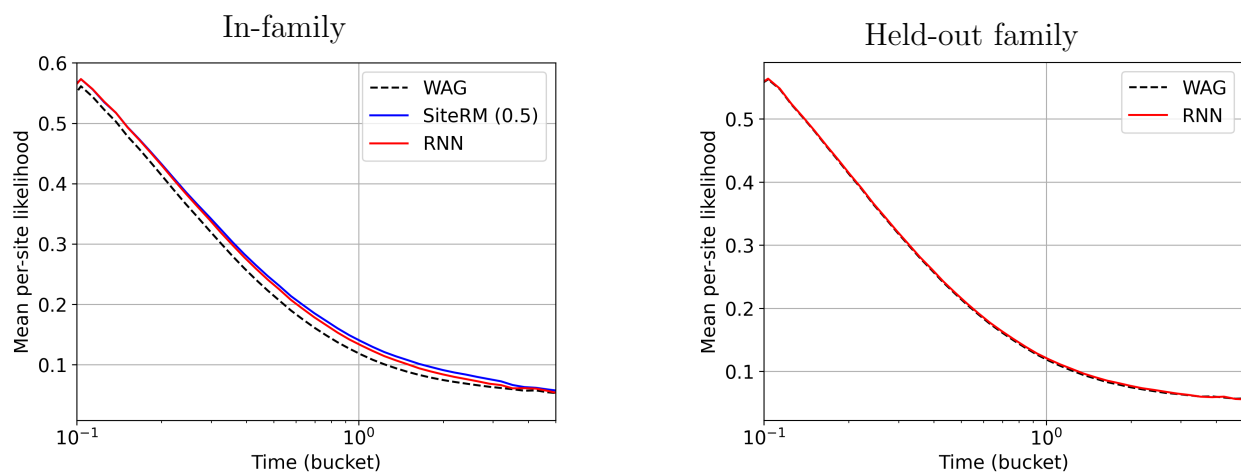


Figure 2.3: In-family and held-out family per-site likelihoods over temporal distance between sequences. The effect of gap characters has been normalized out. The RNN outperforms WAG on in-family transitions but reduces to WAG on held-out transitions. SiteRM has comparable performance to the RNN, sometimes even beating it.

Chapter 3

Ancestral Sequence Reconstruction

We now apply three of the protein evolution models trained in the previous section—WAG, SiteRM, and the RNN—to ancestral sequence reconstruction and compare their performance against IQ-Tree [17], a widely used tool for phylogenetic analysis and reconstruction that supports a variety of evolution models and settings.

3.1 Datasets

Problems with Previous Evaluation Methods

Despite the profusion of protein evolution models available to users seeking to perform ASR, there exist relatively few reliable methods for assessing the true performance of ASR methods. This problem stems from the difficulty in obtaining phylogenetic trees with access to ground truth states for ancestral nodes.

Computational studies have largely focused on benchmarking performance against simulated data [19]. However, this can bias the dataset depending on the evolution model used to simulate said data [30]. Predictably, the model used to simulate the data will often have the best performance on it. Many of these simulated datasets can be very simplistic in nature with maximum parsimony outperforming many state-of-the-art methods.

Previous studies have more often sought to achieve faster and more accurate estimation of rate matrices under known methods, thus allowing them to directly compare likelihoods of trees and how often their method is selected as the best-fit model under existing toolkits [17, 18].

Only a single experimentally determined dataset (via directed evolution) is easily accessible [24], but its extant samples are extremely limited in sequence divergence. Moreover, concerns have been raised over its relevance to natural evolution and applicability to real biological systems that have evolved on a geological timescale [29].

We tried to derive an experimental viral dataset from COVID and influenza samples collected in the wild using Nextstrain [11], but available samples were too low in sequence

divergence (at most 10% in the cases we investigated) to perform effective ASR.

As such, the predominant ASR evaluation methodology which relies on simulated dataset raises problems when attempting to benchmark our new protein evolution models. For example, the SiteRM model [22] has many more parameters than the models used to simulate these datasets. The simulation methods also operate on an independent sites model while our RNN model does not so coevolving sites and other naturally arising properties may not be effectively captured. In order to avoid the potential problem of evaluating our models on data simulated with potentially inferior models of evolution, we turn to a recently proposed method by Sennett et Theobald known as extant sequence reconstruction (ESR) [29].

Extant Sequence Reconstruction

ESR enables the evaluation of ASR methods on any dataset, needing only an MSA over the extant sequences and optionally the tree topology. There is no need to have access to the ground true ancestral sequences.

The basic idea is rather simple. Instead of reconstructing the state of an unknown ancestral node, we remove an extant sequence (a leaf) from the dataset and use an ASR method to recover its sequence. Since we know its true sequence, we have a ground truth to evaluate against. The central idea is illustrated in Figure 3.1.

The procedure is justified by the time-reversible assumption underlying the phylogenetic tree (used by all known existing ASR and tree reconstruction methods). Also known as Felsenstein’s pulley principle [7], this property means that there is effectively no distinction between ancestor and descendant. Thus the tree can be re-rooted at a leaf node of choice and we can perform ASR with the leaf as the ancestral node to reconstruct. In this way ESR can be thought of as a kind of sequence-level cross-validation. Henceforth ESR and ASR may be used interchangeably, since ESR is just ASR on a re-rooted tree and we have no true ancestral sequence datasets.

It is also worth nothing that ESR holds merit of its own when considering viral evolution. For example, scientists might be interested in predicting the characteristics of next year’s influenza strain. Reconstructing a new extant sequence given previous ones is exactly the problem ESR aims to solve.

Evaluation Data

With ESR as our evaluation method, we are able to continue using the trRosetta dataset for which we have already estimated trees with branch lengths. We selected 13 families from the 1,000 used to train the RNN, WAG, and SiteRM models. We used the test trees associated with these 13 families to benchmark the performance of these 3 models on ASR, using 25 iterations of extant selection.

We weren’t able to benchmark on all 1,000 families due to runtime limitations but hope to work with larger datasets in future work. While reconstructing the root of a given tree is linear in the sequence length and number of nodes, there remain large constant factors that

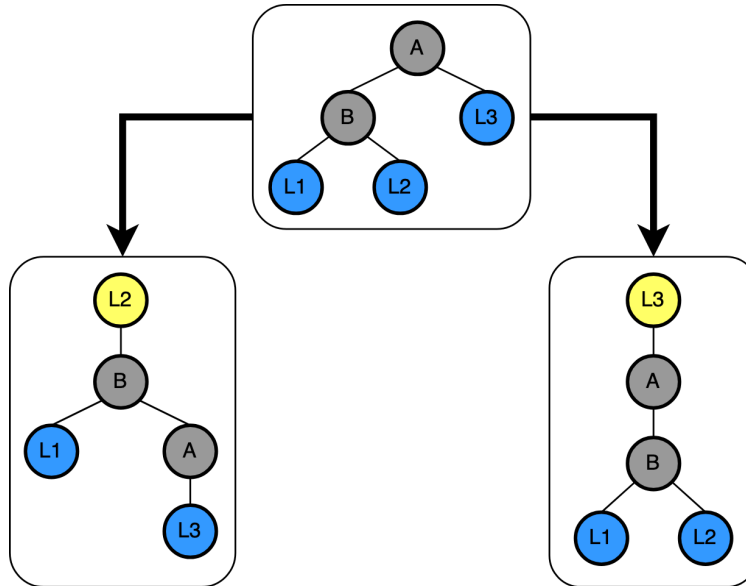


Figure 3.1: Illustration of two iterations of extant sequence reconstruction (ESR). Gray nodes are internal without ground truth sequences, blue nodes are leaves in the original tree (top) with ground truth sequences, and yellow nodes are the extant sequences to be inferred during ESR. In the iteration shown on the left, we select a random leaf node (L2) from the original tree and make it the new root. The new root’s sequence is inferred from the remaining leaves (L1 and L3 in this case). A similar procedure occurs on the right with L3 being the selected leaf in that iteration. Since both L2 and L3 were originally leaves, we can evaluate against their ground truth sequences.

need to be optimized on a method-by-method basis (for example, reconstructing different parts of the tree in parallel). Such optimizations were not undertaken in this project due to the large number of methods we initially tried to test.

The 13 particular families which we evaluated on were chosen because they coincided with the set of families used in a previous work [19] whose performance we hoped to compare against. Prior ASR works often evaluate on small numbers of families [19, 24, 1] (and even smaller trees than ours), so we believe 13 families is an acceptable amount.

Unfortunately, we found too many issues with the methodology of the paper [19] we intended to compare against to use it as an accurate comparison. The simulation settings used to generate the trees were general and not sufficiently tuned to the specific families. Moreover, our implementation of maximum parsimony consistently outperformed all other methods benchmarked in the paper (the original paper did not show parsimony results). This should not be the case as a great deal of work has shown that parsimony, while sometimes competitive, generally loses out to likelihood based methods on realistic trees [28].

A summary of the datasets and families chosen for evaluation is shown below in Table 3.1. The third column indicates the average percentage of the alignment which is composed

of gap characters and the fourth column is the average branch length in the test tree.

Table 3.1: Summary statistics of ASR evaluation datasets. Shows the average percentage of gaps in each sequence of the family’s training alignment as well as the average transition time across those train samples.

| trRosetta ID | Alignment Length | Avg. % of Gaps | Average Transition | Description |
|--------------|------------------|----------------|--------------------|--|
| 5leo | 93 | 28.28 | 0.6094 | Bacterial SH3 domain |
| 3l11 | 104 | 19.08 | 0.5251 | Zinc finger, C3HC4 type |
| 3gxx | 156 | 19.52 | 0.6034 | Beta-lactamase hydrolase-like protein |
| 4tse | 157 | 18.25 | 0.6410 | Mind bomb SH3 repeat domain |
| 1c25 | 161 | 38.19 | 0.8591 | Rhodanese-like domain |
| 5hd9 | 194 | 19.43 | 0.5857 | Podovirus DNA encapsidation protein |
| 6gap | 213 | 24.12 | 0.6793 | Reovirus viral attachment protein sigma 1 |
| 2d4c | 231 | 6.696 | 0.4712 | BAR domain |
| 1raj | 257 | 5.666 | 1.014 | Viral RNA-dependent RNA polymerase |
| 2hjh | 325 | 13.69 | 0.4764 | Sir2 family |
| 4be2 | 368 | 13.00 | 0.7464 | Retroviral integrase C-terminal SH3 domain |
| 5tip | 436 | 12.96 | 0.4733 | Eukaryotic DNA virus major capsid protein |
| 3hky | 564 | 34.51 | 0.9168 | Viral RNA dependent RNA polymerase |

3.2 Methods

We autoregressively decode each character of y in sequential order. As a simple example consider a tree with root y connected to a single leaf x by an edge of length t (there are only two nodes in this tree). To decode the i th character of y , our first method tries optimizing the following expression:

$$\arg \max_{y_i} P(y_i | x_i, y[1 : i - 1], t)$$

where the probability function $P(\cdot)$ is further parameterized by a particular protein evolution model (WAG, SiteRM, or RNN). Note that under an independent sites model the prior context of x and y are not used, so the inner expression simplifies to $P(y_i | x_i, t)$. Under the RNN, this previous context matters, however.

We call this expression the forward likelihood since it simulates maximizing the likelihood of transitioning from a known state x_i into the unknown state y_i . We also try optimizing what we will refer to as the backwards likelihood:

$$\arg \max_{y_i} P(x_i | y[1 : i - 1], y_i, t)$$

which is analogous to transitioning *from* the unknown state y_i into x_i . In practice our models perform much better with the backwards likelihood, and from now on whenever not specified we use the backwards likelihood to optimize the transition between two sequences.

We can go further by multiplying with a prior. As all of our models fundamentally use a rate matrix approach, we use the stationary distribution of the rate matrix for that position as the prior:

$$\arg \max_{y_i} (\pi(y_i)P(x_i|y[1 : i - 1], y_i, t))$$

which is essentially maximum a posteriori estimation. We found that using the prior marginally improved results and so use it for all our analyses.

Composite Likelihood Over Leaves

Our first method reconstructs the sequence of the root of the tree y from only the n sequences of the leaves $x^{(1)}, x^{(2)}, \dots, x^{(n)}$ and their corresponding transition times t_1, t_2, \dots, t_n . Each transition time t_i is exactly the sum of the branch lengths along the path from y to the i th leaf.

We optimize a composite log likelihood over all the n leaves of the tree to reconstruct the root character by character. Letting $x_i^{(j)}$ be the i th character of the j th leaf's sequence we have:

$$\arg \max_{y_i} \sum_{j=1}^n \log (\pi(y_i)p(x_i^{(j)}|y[1 : i - 1], y_i, t_i))$$

where we repeat the procedure character by character from $i = 1$ to $i = L$. Like before, only the RNN model requires the context of previously predicted tokens of $y[1 : i - 1]$.

Recursive Strategies

A full composite likelihood over all the leaves runs into the potential problem of transition times that are too long since the leaves can be very far from the root. Transitioning from such a large number of leaves also tends to generate more gappy sequences since many models seem to treat the gap character as an unknown state (so all leaves contribute to the gap state while each contributes to potentially different non-gap states). The second set of approaches we tried were a selection of recursive strategies that decreased both the number of terms in the likelihood and the length of the transitions.

First we tried a fully recursive strategy in which we reconstruct each node from its direct children. If needed, we recursively reconstruct the children. Thus we end up reconstructing the tree from the bottom up (nodes that are parents of leaves, then parents of those nodes, etc.).

We then tried reconstructing each node from descendants which were not its direct children. Specifically, we choose a minimum distance T and traverse the tree from the unknown node to find its closest descendants that are at least T away. The node is reconstructed from those descendants, with the procedure recursing as before in case the sequence of any of those descendants is unknown. This ended up improving our performance as well as speeding up the runtime of the algorithm.

Setting T equal to twice the average branch works well in practice, perhaps due to the similarity between these transition times and the training transition times which were drawn from cherries.

Initialization With Parsimony

With many of the above methods we found models tending to produce excessive numbers of gap characters. Though this was in part alleviated by shifting from forward likelihood decoding to backwards likelihood decoding, reconstructed roots were still much often shorter in length than desired. To address this problem we tried a variety of strategies that initialized states with parsimony (which already worked quite well).

One method iteratively updates sequences in the tree from their surrounding nodes. With maximum parsimony states available for all nodes, we are able to include both descendants and ancestors in the composite likelihood transition function. Some attempted strategies included using all nodes along a path to the unknown node in the composite likelihood or using a nodes direct ancestors (up to a certain distance) as well as its descendants. This iteration would be performed repeatedly on all nodes in the tree in a random order. Unfortunately, this method performed quite poorly in practice and often even decreased the fidelity of the reconstructed root. Without access to ground truth states for internal nodes we were unable to assess whether the procedure improved their reconstructed states which would be a major motivation for this method.

Our best performing strategy was to use the locations of gaps as predicted by maximum parsimony as input to the decoding process. More specifically, we constrained models to predict a gap character in a particular position if the maximum parsimony reconstruction had already predicted a gap in that position. If maximum parsimony did not have a gap at a position, the model was constrained to predicting a non-gap character.

3.3 Results

We benchmark against IQ-Tree, a widely used tool for phylogenetic analysis and maximum likelihood reconstruction, as our SOTA benchmark [17].

Our test trees all contain around 500 leaves each. However, IQ-Tree works best with a number of sequences equal to half the alignment length; the program warns the user that the problem may be overparameterized if too many sequences are used and requests that the MSA size be reduced.

To fix this, we also evaluate reconstruction on a version of the tree with subsampled leaves. For example, in the 3hky dataset with an alignment length of 564 leaves we evaluate on a version of the tree in which we prune all but 282 leaves. The kept leaves either consist of the set of furthest leaves or the set of leaves in the middle of the list of leaves sorted by distance from the root.

This results in us having two “difficulties” of each tree to reconstruct: one using the IQ-Tree optimal number of leaves at the furthest distance (`furthest_leaves`) and one using the optimal number of leaves at a medial distance (`middle_leaves`). We choose not to use the version of the tree with all leaves (or the closest leaves) since the problem becomes much easier with our models being able to just predict the root from the closest leafs (and ignore the others that are further away).

Following prior ASR works [24, 19, 1], we use average percent identity as our method for evaluating the correctness of reconstructed roots (extant sequences in ESR). To calculate this we divide the number of correctly reconstructed sites by the total number of sites.

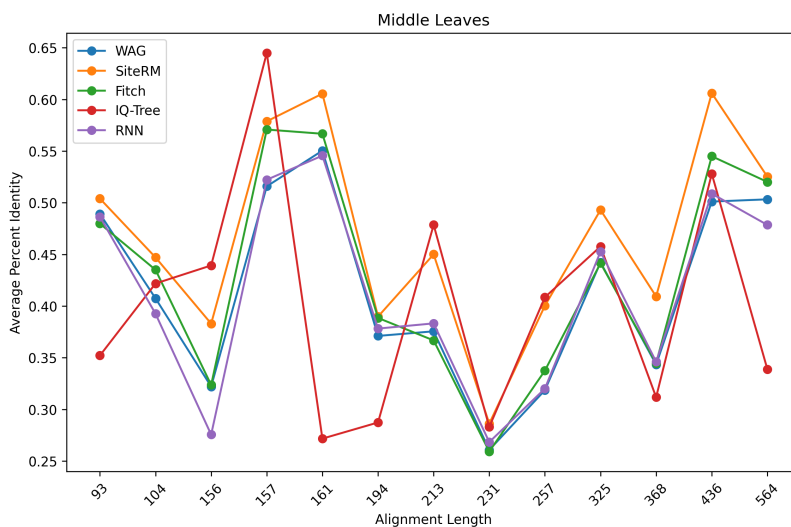
We found that using backwards likelihood decoding and stationary distribution priors categorically improved reconstructions. Likewise, using maximum parsimony initialization for gap states dramatically mitigated the tendencies of models like the RNN to predict too many gaps. Further, the decoding method using descendants at a distance of at least twice the average branch length from the unknown node worked best (also utilizing these techniques of backwards likelihood, priors, and fitch initialization).

Each datapoint was computed over 25 ESR iterations, with each iteration being the reconstruction of a tree with a different leaf at the root. The final results for our models over both difficulties are shown below in Figures 3.2a and 3.2b.

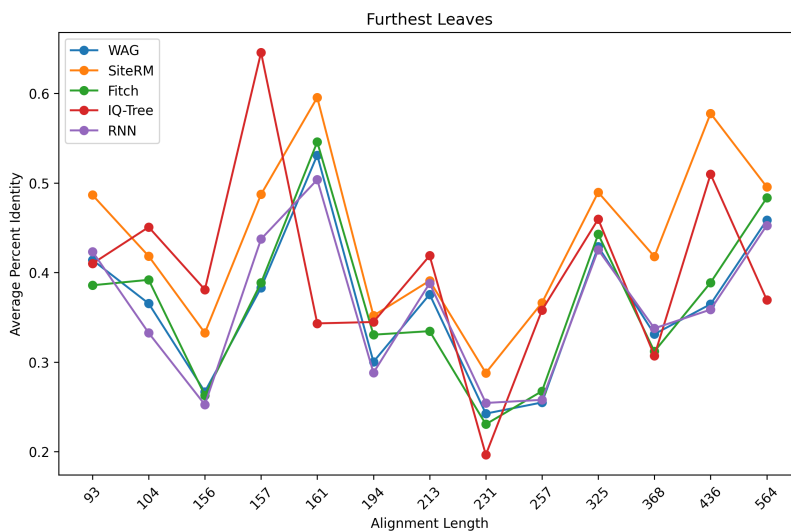
Comparing the two plots, we can see that models generally perform better when using leaves closer to the root (middle leaves) than those further away (furthest leaves) as expected. Our SiteRM model generally outperforms the state-of-the-art IQ-Tree, though it does seem to lose out on shorter alignments. Its consistent performance over IQ-Tree in the longer alignment sequences is worth noting (325, 368, 436, and 564). On the other hand, the RNN unfortunately seems to collapse back down to WAG’s performance.

Furthermore, maximum parsimony (Fitch’s algorithm) performs very well and quite often beats WAG and sometimes even IQ-Tree. Surprising as this might be, it turns out that maximum parsimony generally performs quite well in ASR literature. In some paper we attempted to replicate [19], we found that maximum parsimony outperformed all listed methods benchmarked in that paper (though the work did not itself include maximum parsimony as a method). In another [24] maximum parsimony performed on par with other state-of-the-art methods.

There does not seem to be an observable trend between alignment length and accuracy of the reconstruction, which could speak to the highly specialized nature of ASR. Namely, that the molecular properties of the protein family being reconstructed might matter more than superficial characteristics like the sequence length. Of course, this could also be an issue with our evaluation methodology and so merits further investigation regardless.



(a) Reconstruction over leaves in the middle of the tree.



(b) Reconstruction over leaves at the bottom of the tree.

Figure 3.2: Reconstructing extant root over 25 ESR iterations using a number of leaves equal to half the alignment length with the method using descendants at twice the average branch length away. Leaf selection is either those in the middle of the tree or those furthest from the root.

Chapter 4

Conclusion

4.1 Discussion

Both the RNN [14] and SiteRM [22] models show increased performance over the classical WAG model when compared against single sequence-to-sequence transitions. Although the RNN seems to perform no better than WAG on ASR and only the SiteRM model's performance carries over, we have shown that we can build deep models that strictly generalize classical models. It is possible that the RNN could perform better than WAG on other downstream tasks.

Nevertheless, we have shown that SiteRM model with relatively simple reconstruction algorithms can generally outperform the state-of-the-art program IQ-Tree on the families in our evaluation set. Moreover, the consistently higher performance of SiteRM in the longer sequence length domain shows promise.

4.2 Future Work

First and foremost, future work will require much more care and effort in choosing evaluation datasets. In the early stages of the project we had intended to use datasets from prior ASR papers [19, 24], but after working with them for some time found them to be either disastrously biased or very limited in sequence divergence. Searching for varied, model-independent or sufficiently expressive datasets is a challenging task but one which will greatly improve confidence in benchmark results.

Otherwise if continuing with ESR, more attention needs to be paid to the particular families chosen. It is difficult to obtain consistent measurements when the reconstruction performance across different families varies so much. While it is possible that this is simply caused by the arrangement/prevalence of gaps in the different alignments, it might also be caused by some molecular properties of the sequences that are unable to be captured in our current framework. If nothing else, we should at least evaluate on a much larger set of families which will require code optimizations to improve runtime.

There is also a dire need for better metrics than average percent identity. Although this is the approach used by nearly all the prior ASR work we have investigated, it need not correlate with the efficacy or functional performance of the reconstructed protein. More biologically accurate evaluation metrics could allow for other more complex methods like posterior decoding to show their worth (posterior decoding has been shown in some capacity to improve stability of the reconstructed protein, but this is not captured by percent identity).

While we could continue to iterate on the RNN to see if its performance can pass that of WAG on ASR (for example, initializing with SiteRM site-specific rate matrices), future work should likely be devoted to exploring more expressive deep models of protein evolution. Attention-based models will be a much stronger movement away from independent-sites models and, given their exceptional performance on other tasks like NLP, likely do a much better job than any RNN. Leveraging pre-trained models like ESM [27] could also do away with the need for including families of interest in the train-set.

Lastly, there is a strong case to be made for moving away from gap-aware models (reverting from 21 by 21 rate matrices to 20 by 20 matrices). Our treatment of gaps as an additional character presented problems during ASR with the best solution being to not predict them at all (and instead use maximum parsimony to predict gap states). The tendency for models to treat gaps as a kind of missing data instead of as an indel event presented significant problems in ASR.

Bibliography

- [1] Naila O. Alieva et al. “Diversity and Evolution of Coral Fluorescent Proteins”. In: *PLOS ONE* 3.7 (July 2008), pp. 1–12. DOI: 10.1371/journal.pone.0002680. URL: <https://doi.org/10.1371/journal.pone.0002680>.
- [2] Miguel Arenas. “Protein Evolution in the Flaviviruses”. In: *Journal of Molecular Evolution* 88.6 (2020), pp. 473–476. ISSN: 1432-1432. DOI: 10.1007/s00239-020-09953-1. URL: <https://doi.org/10.1007/s00239-020-09953-1>.
- [3] Belinda S. W. Chang et al. “Recreating a Functional Ancestral Archosaur Visual Pigment”. In: *Molecular Biology and Evolution* 19.9 (2002), pp. 1483–1489. ISSN: 0737-4038. DOI: 10.1093/oxfordjournals.molbev.a004211. URL: <https://doi.org/10.1093/oxfordjournals.molbev.a004211>.
- [4] M. O. Dayhoff and R. M. Schwartz. “Chapter 22: A model of evolutionary change in proteins”. In: *Atlas of Protein Sequence and Structure* (1978).
- [5] Dr. Lewis Pirenne. *ASR phylogeny*. Accessed: 2024-08-06. 2021. URL: https://commons.wikimedia.org/wiki/File:ASR_phylogeny.png.
- [6] Mariette F. Ducatez et al. “Feasibility of reconstructed ancestral H5N1 influenza viruses for cross-clade protective vaccine development”. In: *Proceedings of the National Academy of Sciences* 108.1 (2011), pp. 349–354. DOI: 10.1073/pnas.1012457108. eprint: <https://www.pnas.org/doi/pdf/10.1073/pnas.1012457108>. URL: <https://www.pnas.org/doi/abs/10.1073/pnas.1012457108>.
- [7] Joseph Felsenstein. “Evolutionary trees from DNA sequences: A maximum likelihood approach”. In: *Journal of Molecular Evolution* 17.6 (1981), pp. 368–376. DOI: 10.1007/BF01734359. URL: <https://doi.org/10.1007/BF01734359>.
- [8] Walter M. Fitch. “Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology”. In: *Systematic Biology* 20.4 (Dec. 1971), pp. 406–416. ISSN: 1063-5157. DOI: 10.1093/sysbio/20.4.406. eprint: <https://academic.oup.com/sysbio/article-pdf/20/4/406/4697394/20-4-406.pdf>. URL: <https://doi.org/10.1093/sysbio/20.4.406>.

- [9] Stéphane Guindon et al. “New Algorithms and Methods to Estimate Maximum-Likelihood Phylogenies: Assessing the Performance of PhyML 3.0”. In: *Systematic Biology* 59.3 (May 2010), pp. 307–321. DOI: 10.1093/sysbio/syq010. eprint: <https://academic.oup.com/sysbio/article-pdf/59/3/307/24207259/syq010.pdf>. URL: <https://doi.org/10.1093/sysbio/syq010>.
- [10] Yosephine Gumulya and Elizabeth M.J. Gillam. “Exploring the past and the future of protein evolution with ancestral sequence reconstruction: the ‘retro’ approach to protein engineering”. In: *Biochemical Journal* 474.1 (Dec. 2016), pp. 1–19. ISSN: 0264-6021. DOI: 10.1042/BCJ20160507. eprint: <https://portlandpress.com/biochemj/article-pdf/474/1/1/687940/bcj-2016-0507c.pdf>. URL: <https://doi.org/10.1042/BCJ20160507>.
- [11] James Hadfield et al. “Nextstrain: real-time tracking of pathogen evolution”. In: *Bioinformatics* 34.23 (2018), pp. 4121–4123. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty407. eprint: https://academic.oup.com/bioinformatics/article-pdf/34/23/4121/48921142/bioinformatics_34_23_4121.pdf. URL: <https://doi.org/10.1093/bioinformatics/bty407>.
- [12] Natalie M. Hendrikse et al. “Exploring the therapeutic potential of modern and ancestral phenylalanine/tyrosine ammonia-lyases as supplementary treatment of hereditary tyrosinemia”. In: *Scientific Reports* 10.1 (2020), p. 1315. ISSN: 2045-2322. DOI: 10.1038/s41598-020-57913-y. URL: <https://doi.org/10.1038/s41598-020-57913-y>.
- [13] David T. Jones, William R. Taylor, and Janet M. Thornton. “The rapid generation of mutation data matrices from protein sequences”. In: *Bioinformatics* 8.3 (June 1992), pp. 275–282. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/8.3.275. eprint: <https://academic.oup.com/bioinformatics/article-pdf/8/3/275/479648/8-3-275.pdf>. URL: <https://doi.org/10.1093/bioinformatics/8.3.275>.
- [14] Koehl, A.* et al. “Deep Models of Protein Evolution in Time”. Preprint. *Equal contribution; authors listed alphabetically. 2024.
- [15] Jeffrey M. Koshi and Richard A. Goldstein. “Probabilistic reconstruction of ancestral protein sequences”. In: *Journal of Molecular Evolution* 42 (1996), pp. 313–320. URL: <https://api.semanticscholar.org/CorpusID:15997589>.
- [16] Si Quang Le, Cuong Cao Dang, and Olivier Gascuel. “Modeling Protein Evolution with Several Amino Acid Replacement Matrices Depending on Site Rates”. In: *Molecular Biology and Evolution* 29.10 (Apr. 2012), pp. 2921–2936. ISSN: 0737-4038. DOI: 10.1093/molbev/mss112. eprint: <https://academic.oup.com/mbe/article-pdf/29/10/2921/3416403/mss112.pdf>. URL: <https://doi.org/10.1093/molbev/mss112>.

- [17] Bui Quang Minh et al. “IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era”. In: *Molecular Biology and Evolution* 37.5 (Feb. 2020), pp. 1530–1534. ISSN: 0737-4038. DOI: 10.1093/molbev/msaa015. eprint: <https://academic.oup.com/mbe/article-pdf/37/5/1530/33386032/msaa015.pdf>. URL: <https://doi.org/10.1093/molbev/msaa015>.
- [18] Bui Quang Minh et al. “QMaker: Fast and Accurate Method to Estimate Empirical Models of Protein Evolution”. In: *Systematic Biology* 70.5 (Feb. 2021), pp. 1046–1060. ISSN: 1063-5157. DOI: 10.1093/sysbio/syab010. eprint: <https://academic.oup.com/sysbio/article-pdf/70/5/1046/39679725/syab010.pdf>. URL: <https://doi.org/10.1093/sysbio/syab010>.
- [19] Lys Sanz Moreta et al. “Ancestral protein sequence reconstruction using a tree-structured Ornstein-Uhlenbeck variational autoencoder”. In: *International Conference on Learning Representations*. 2022. URL: <https://openreview.net/forum?id=FZoZ7a31GCW>.
- [20] Morgan N. Price, Paramvir S. Dehal, and Adam P. Arkin. “FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments”. In: *PLOS ONE* 5.3 (Mar. 2010), pp. 1–10. DOI: 10.1371/journal.pone.0009490. URL: <https://doi.org/10.1371/journal.pone.0009490>.
- [21] Sebastian Prillo et al. “CherryML: scalable maximum likelihood estimation of phylogenetic models”. In: *Nature Methods* 20.8 (2023), pp. 1232–1236. ISSN: 1548-7105. DOI: 10.1038/s41592-023-01917-9. URL: <https://doi.org/10.1038/s41592-023-01917-9>.
- [22] Prillo, S.*, Wu, W.*, and Y.S. Song. “Ultrafast classical phylogenetic method beats large protein language models on variant effect prediction”. Preprint. *Equal contribution; authors listed alphabetically. 2024.
- [23] Tal Pupko et al. “A Fast Algorithm for Joint Reconstruction of Ancestral Amino Acid Sequences”. In: *Molecular Biology and Evolution* 17.6 (2000), pp. 890–896. ISSN: 0737-4038. DOI: 10.1093/oxfordjournals.molbev.a026369. URL: <https://doi.org/10.1093/oxfordjournals.molbev.a026369>.
- [24] Ryan Randall et al. “An experimental phylogeny to benchmark ancestral sequence reconstruction”. In: *Nat Commun* 7 (Sept. 2016). DOI: 10.1038/ncomms12847.
- [25] Liam J. Revell. “ANCESTRAL CHARACTER ESTIMATION UNDER THE THRESHOLD MODEL FROM QUANTITATIVE GENETICS”. In: *Evolution* 68.3 (2014), pp. 743–759. DOI: <https://doi.org/10.1111/evo.12300>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/evo.12300>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/evo.12300>.
- [26] Valeria A. Risso et al. “Hyperstability and Substrate Promiscuity in Laboratory Resurrections of Precambrian β -Lactamases”. In: *Journal of the American Chemical Society* 135.8 (2013), pp. 2899–2902. ISSN: 0002-7863. DOI: 10.1021/ja311630a. URL: <https://doi.org/10.1021/ja311630a>.

- [27] Alexander Rives et al. “Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences”. In: *Proceedings of the National Academy of Sciences* 118.15 (2021). bioRxiv 10.1101/622803, e2016239118. DOI: 10.1073/pnas.2016239118. URL: <https://www.pnas.org/doi/full/10.1073/pnas.2016239118>.
- [28] Avery G. A. Selberg, Eric A. Gaucher, and David A. Liberles. “Ancestral Sequence Reconstruction: From Chemical Paleogenetics to Maximum Likelihood Algorithms and Beyond”. In: *Journal of Molecular Evolution* 89.3 (2021), pp. 157–164. ISSN: 1432-1432. DOI: 10.1007/s00239-021-09993-1. URL: <https://doi.org/10.1007/s00239-021-09993-1>.
- [29] Michael A. Sennett and Douglas L. Theobald. “Extant Sequence Reconstruction: The Accuracy of Ancestral Sequence Reconstructions Evaluated by Extant Sequence Cross-Validation”. In: *Journal of Molecular Evolution* 92.2 (2024), pp. 181–206. ISSN: 1432-1432. DOI: 10.1007/s00239-024-10162-3. URL: <https://doi.org/10.1007/s00239-024-10162-3>.
- [30] Alexandros Stamatakis. “RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies”. In: *Bioinformatics* 30.9 (Jan. 2014), pp. 1312–1313. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btu033. eprint: https://academic.oup.com/bioinformatics/article-pdf/30/9/1312/48923136/bioinformatics_30_9_1312.pdf. URL: <https://doi.org/10.1093/bioinformatics/btu033>.
- [31] Simon Whelan and Nick Goldman. “A General Empirical Model of Protein Evolution Derived from Multiple Protein Families Using a Maximum-Likelihood Approach”. In: *Molecular Biology and Evolution* 18.5 (May 2001), pp. 691–699. ISSN: 0737-4038. DOI: 10.1093/oxfordjournals.molbev.a003851. eprint: <https://academic.oup.com/mbe/article-pdf/18/5/691/23447821/i0737-4038-018-05-0691.pdf>. URL: <https://doi.org/10.1093/oxfordjournals.molbev.a003851>.
- [32] Jianyi Yang et al. “Improved protein structure prediction using predicted inter-residue orientations”. In: *bioRxiv* (2019). DOI: 10.1101/846279. eprint: <https://www.biorxiv.org/content/early/2019/11/18/846279.full.pdf>. URL: <https://www.biorxiv.org/content/early/2019/11/18/846279>.
- [33] Ziheng Yang, Sudhir Kumar, and Masatoshi Nei. “A New Method of Inference of Ancestral Nucleotide and Amino Acid Sequences”. In: *Genetics* 141 (Dec. 1995), pp. 1641–50. DOI: 10.1093/genetics/141.4.1641.
- [34] Philip M. Zakas et al. “Enhancing the pharmaceutical properties of protein drugs by ancestral sequence reconstruction”. In: *Nature Biotechnology* 35.1 (2017), pp. 35–37. ISSN: 1546-1696. DOI: 10.1038/nbt.3677. URL: <https://doi.org/10.1038/nbt.3677>.

- [35] Peng Zhou et al. “A pneumonia outbreak associated with a new coronavirus of probable bat origin”. In: *Nature* 579.7798 (2020), pp. 270–273. ISSN: 1476-4687. DOI: 10.1038/s41586-020-2012-7. URL: <https://doi.org/10.1038/s41586-020-2012-7>.