Nested birth-death processes are competitive with parameter-heavy neural networks as time-dependent models of protein evolution

Annabel Large [1 2] **Ian Holmes** [1 2]

Abstract

Most statistical phylogenetics analyses use relatively simple continuous-time finite-state Markov models of point substitution to describe molecular evolution, often keeping sequence length fixed, ignoring insertions and deletions (indels) entirely, and making little (if any) allowance for variations in selection pressure due to interactions between amino acids. The simplistic assumptions of these models limit the realism of phylogenetics. We extend the TKF92 model - the canonical hierarchical model combining an outer birth-death process for indels with an inner finite-state Markov chain for substitutions - by introducing additional nesting and latent states, allowing for structural heterogeneity. We compare these TKF92 extensions (which are derived as exact solutions of instantaneous processes, and in which evolutionary time naturally appears as a matrix exponential coefficient) to two classes of neural seq2seq model that are not derived in such a way, but instead take evolutionary time as an input feature: the first class of model being constrained to enforce a TKF92-like structure, and the second lacking any such constraint. We evaluate the per-character perplexities of all models on splits of the PFam database of aligned protein domains. A nested TKF-based model with only 32,000 parameters is highly competitive with neural networks containing tens of millions of parameters, outperforming all but two of the neural architectures tested. Our results indicate that approaches grounded in molecular evolutionary theory may be more parameter-efficient and provide a better fit to real alignments than unconstrained alternatives, supporting the incorporation of CTMC-based model structure within future neural phylogenetic approaches.

[1]Department of Bioengineering, University of California, Berkeley, California, United States [2]Center for Computational Biology, University of California, Berkeley. Correspondence to: Annabel Large <annabel_large@berkeley.edu>.

1. Introduction

Substitutions and insertion-deletion (indel) events can alter protein function and drive complex evolutionary phenomena. Improved understanding of sequence evolution through time-dependent statistical models benefits areas such as epidemiology, biotechnology, and population genetics.

TKF91 and TKF92 are the canonical indel models for substitutions and indels, assuming an underlying instantaneous rate process (Thorne et al., 1991; 1992). These models take the form of hierarchically nested continuous-time Markov chains (CTMCs). At the outer level, a linear birth-death process over "links" describes changes in sequence length. At the inner level, each residue independently evolves under a finite-state CTMC. Exact finite-time solutions to these processes yield transition and emission parameters in a hidden Markov model (HMM) that defines the autoregressive likelihood $\mathrm{P}(Z, Y|X, t)$. Here Z is an alignment between ancestral sequence X and Y, assuming a specified evolutionary drift time t. Other related HMM-based indel models derive principled approximations to the underlying indel process (Löytynoja & Goldman, 2005; Redelings & Suchard, 2007; De Maio, 2021; Holmes, 2020).

In reality, each residue in a protein may experience different selection pressures due to biophysical constraints. Such rate variation can be accommodated in solvable CTMC-based models by composition, albeit in limited ways. For example, heterogeneity in substitution rate and mutation biases is routinely accounted for by introducing latent position-varying parameters into the model (Yang, 1994; Le & Gascuel, 2008; Prillo et al., 2024). Indel models are similarly amenable to such extensions and compositions (Holmes, 2004).

Use of these solvable CTMC-based models has both benefits and drawbacks. Benefits include directly interpretable rate parameters that quantify evolutionary dynamics, and that missing or artefactual data (e.g. sequence alignments) can be marginalized out of the likelihood. Drawbacks include that solvable CTMCs are limited in the complexity of the phenomena they can model. For example, indel rates are restricted by local sequence context and longer-range epistatic interactions, but incorporating this into a CTMC-based model is challenging.

An alternative to deriving HMM-based solutions is to use an autoregressive neural network to approximate the same empirical finite-time distribution $\mathrm{P}(Z, Y|X, t)$. This would increase expressive power, as neural architectures can capture complex multi-residue interactions, both local and distal, in ways that compositions of independent, solvable local CTMCs cannot (Wilburn & Eddy, 2020). However, neural parameters cannot be directly interpreted as quantifications of evolutionary dynamics.

In this work, we explore several competing approaches to modeling molecular drift of sequences by substitution and indel events. All models provide an autoregressive likelihood of the form $\mathrm{P}(Z, Y|X, t)$. We curate a dataset of protein domain alignments from the PFam database. We compare several related HMM-based models (Thorne et al., 1991; 1992; Löytynoja & Goldman, 2005; Redelings & Suchard, 2007; Holmes, 2020) based on their cross-entropy on withheld alignments. TKF92 emerges as the best of the simple HMM-based models, so we extend TKF92 via mixtures and compositions at various levels of the hierarchically nested process. The resulting HMM-based elaborations of TKF92 gain expressivity while remaining exactly solvable. Our most expressive HMM is, to our knowledge, the first HMM-based indel model to allow indel rates to depend on local sequence context. We also develop two methods that incorporate neural sequence embeddings. One neural model learns site- and sample-specific substitution and indel rates for a TKF92 model, yielding some interpretability. In contrast, the other uses generic neural network modules without any evolution-specific architectural components. On our Pfam alignment dataset, the models developed as solvable elaborations of TKF92 prove to be competitive with neural methods while using orders of magnitude fewer parameters.

All our proposed methods are constructed to satisfy what we call the "alignment-Markovian" property. The autoregressive likelihood $\mathrm{P}(Z, Y|X, t)$ is constrained to be strictly Markovian with respect to the alignment, though not necessarily with respect to the sequences. This property allows the alignment to be marginalized out, as with TKF92 and other HMM-based models

2. Background and prior work

2.1. Alignment-Markovian models

Let Ω be an alphabet of symbols (e.g. amino acids), and $\Omega_g = \Omega \cup \{\epsilon\}$ be this alphabet augmented with a gap symbol. A pairwise alignment (with all-gap columns excluded) may be considered a string over the alphabet of alignment columns, $\mathcal{A} = (\Omega_g \times \Omega_g) \setminus \{(\epsilon, \epsilon)\}$. Let $A \in \mathcal{A}^*$ be an alignment and let $\mathcal{X} : \mathcal{A}^* \to \Omega^*$ and $\mathcal{Y} : \mathcal{A}^* \to \Omega^*$ be projection functions extracting, respectively, the unaligned ancestor and descendant sequences $X = \mathcal{X}(A)$ and $Y = \mathcal{Y}(A)$.

The models we consider follow an *alignment-Markovian assumption*: the distribution of the next alignment column A_k is restricted to depend on the full ancestral sequence, the already-emitted portion of the descendant sequence, the lapsed evolutionary time, and the gap profile of the previous alignment column A_{k-1} (i.e. whether the ancestor token is a gap, the descendant token is a gap, or neither are gaps). This yields the factorization

$$\mathrm{P}\left(A \mid \mathcal{X}(A), t\right) = \prod_{k=1}^{|A|+1} \mathcal{T}^{Y|X}\left(A_k \mid A_{k-1}, i, j, \mathcal{X}(A), \mathcal{Y}(A_{\ldots k-1}), t\right)$$

with $i = |\mathcal{X}(A_{\ldots k-1})|$ and $j = |\mathcal{Y}(A_{\ldots k-1})|$. The term in the product is the *next-column conditional probability*

$$\mathcal{T}^{Y|X}\left(b \mid a, i, j, X, Y_{\ldots j}, t\right) = \mathrm{P}\left(A_k = b \mid A_{k-1} = a, i, j, \mathcal{X}(A), \mathcal{Y}(A_{\ldots k-1}), t\right)$$

for the next alignment column $b = A_k$ given the previous alignment column $a = A_{k-1}$, current ancestor and descendant sequence positions (i, j), full ancestral sequence context $X = \mathcal{X}(A)$, autoregressive descendant sequence context $Y_{\ldots j} = \mathcal{Y}(A_{\ldots k-1})$, and lapsed evolutionary time t.

An alignment represents an evolutionary hypothesis, rarely observable directly. For given sequences X, Y, the most likely alignment can be inferred with the Viterbi algorithm, or alignment can be marginalized out of the likelihood altogether using the Forward algorithm (Durbin et al., 1998).

2.2. Pair hidden Markov models

The Pair HMM is a special case of an alignment-Markovian model where the next-column probability depends only on the previous alignment column and the next ancestral symbol to be aligned. We use the Moore machine framing of an HMM, where emissions are associated with states, so the next-column conditional probability factors into transition and emission probabilities

$$\mathcal{T}^{Y|X}_{\texttt{HMM}}(b \mid a, X_{i+1}, t) = \mathcal{M}(b, X_{i+1}) T_{\tau(a),\tau(b)} E(b)$$

Here T denotes the time-dependent transition matrix, defined by the indel model; $E(\ldots)$ denotes the time-dependent emissions scoring function, defined by the substitution model and equilibrium distribution; $\mathcal{M}(b, X_{i+1})$ is an indicator function that returns 1 if the given ancestor token is contained in the alignment column, and 0 otherwise; and $\tau(\ldots)$ is the state type of the alignment column, i.e. its gap profile. The allowable state types are match (`M`) for columns containing two non-gap symbols (x, y); insert (`I`) for ancestral gaps aligned to descendant symbols (ϵ, y); delete (`D`) for ancestral symbols aligned to descendant gaps (x, ϵ); and sentinel tokens at sequence end, Start (`S`) and End (`E`).

2.3. F81 and GTR-LG08 point substitution models

TKF92-based models rely on point substitution models to describe evolution at individual sites of the sequence. We consider two forms for these substitution model: F81, a simple model parameterized by its equilibrium distribution (Felsenstein, 1981), and GTR-LG08, a more general, time-reversible model parameterized by an equilibrium distribution and a symmetric exchangeability matrix, which we fix to the LG08 exchangeabilities (Lanave et al., 1984; Le & Gascuel, 2008).

Let $s(t) \sim \text{Subst}(Q, \pi)$ denote the point substitution process, where $s(t)$ is one residue at time t, π is the equilibrium distribution, and Q is the rate matrix (calculated from equilibrium distribution and exchangeabilities). The distribution of descendant symbols in match columns is $\text{P}(y \mid x,\ t) = \exp(Qt)_{xy}$, where $\exp(\ldots)$ denotes the matrix exponential. The distribution of descendant symbols in insert columns is given by the equilibrium distribution.

With respect to the next-column conditional probability, the emissions scoring function $E(b)$ returns the emission probability corresponding to the state type of column b.

2.4. TKF91 and TKF92 indel models: birth-death processes over links and fragments

A point substitution process alone cannot generate sequences with different lengths. TKF91 was the first indel model to describe both point mutations and single-residue indel events in a time-dependent manner (Thorne et al., 1991). However, under TKF91, estimates of indel event counts (and, hence, indel rates) tend to be inflated, since the model only permits one residue to be inserted or deleted per event. Consequently, alignments inferred under TKF91 tend to be unrealistic, with single-character indels scattered all over the sequence instead of being concentrated in a few multi-character indel events. The TKF92 model improves on TKF91 by allowing multiple residues to be inserted or deleted via a single event (Thorne et al., 1992). The nested structure of the TKF models is:

Outer model: The outer model is a linear, continuous-time birth-death process with immigration, where "mortal links" are born with insertion rate λ and die with deletion rate μ. There is also one "immortal link" that can never die. Each link is associated with a nested process, which evolves independently throughout the life of the link. Let $S(t) \sim \text{Links}(\text{M}; \lambda, \mu)$ denote the hierarchically structured *links process*, where $S(t)$ is a sequence at time t and M is the inner stochastic process acting at each mortal link.

TKF91 inner model: In TKF91, each mortal link represents a single residue evolving according to the finite-state CTMC

$$S(t) \sim \text{Links}(\text{Subst}(Q, \pi); \lambda, \mu)$$

for some substitution rate matrix Q and equilibrium π.

TKF92 inner model: In TKF92, each mortal link represents a multi-residue fragment with independently evolving sites. The fragment $s'(t)$ at time t is drawn from a *fragment process* $s'(t) \sim \text{Frag}(\text{M}; r)$ as follows. First a fragment length $K \sim \text{Geometric}(r)$ (with $K \geq 1$) is sampled. The stochastic process corresponding to each character in the fragment is then drawn from the substitution model M. The TKF92-distributed "sequence" of fragments is

$$S(t) \sim \text{Links}(\text{Frag}(\text{Subst}(Q, \pi); r); \lambda, \mu)$$

Both TKF91 and TKF92 are amenable to exact analysis, due to the tractability of the underlying birth-death model (Thorne et al., 1991; 1992). With respect to the next-column conditional probability, closed-form solutions to the TKF processes determine the transition probabilities between column state types, which populate the transition matrix T.

2.5. General Geometric Indel process

The TKF92 model was introduced by Thorne *et al* to remediate perceived problems with the TKF91 model: namely, its failure to model multi-character indel events. TKF92 solves this problem at the cost of introducing latent information into the sequence in the form of fragment boundaries. As a result, the underlying state space of TKF92 is not the set of sequences over the residue alphabet, but the set of sequences of multi-character fragments. Fragment boundaries are never observed, as they are technical conveniences that have no material interpretation. They must be imputed or (preferably) marginalized in any practical analysis.

This issue with TKF92 led several previous researchers to investigate CTMC models whose state consisted solely of a character sequence, with no hidden fragment structure. The canonical example is the General Geometric Indel (GGI) process (De Maio, 2021). This process assumes that indels occur at uniform random positions in the sequence and have geometrically distributed lengths, which corresponds to the maximum entropy assumptions for expected indel rates and lengths. The TKF92 model may be considered as an approximation to the GGI process; for the special case where indels are restricted to single residues, the GGI process reduces exactly to the TKF91 model.

Unlike TKF92 or TKF91, no closed-form solution is available for the finite-time transition probabilities of the GGI process. However, it is straightforward to simulate trajectories from the this process, and one can then compare the observed gap length distributions from these simulations with various approximations and heuristics. Several such approximations have been proposed and evaluated on simulated data (Knudsen & Miyamoto, 2003; Löytynoja & Goldman, 2005; Redelings & Suchard, 2007; De Maio, 2021; Holmes, 2020). Of these approaches, the renormal-

ized ODE approach of (Holmes, 2020), which we refer to as H20, most accurately approximates the distributions of gap lengths observed in GGI-based simulations; it is closely followed by TKF92.

As part of our efforts to develop the theory of indel models beyond TKF92, we assessed these approaches not just as approximations to the idealized GGI process, but as explanatory models of real protein alignments. Along with H20, TKF91, and TKF92, we considered two other models, LG05 (Löytynoja & Goldman, 2005) and RS07 (Redelings & Suchard, 2007), both of which can be characterized as informed guesses at closed-form solutions.

2.6. Mixture of site classes

Using mixture distributions is the standard approach to modeling heterogeneous selection pressures across sites. (Yang, 1994; Le et al., 2008). Instead of being drawn from one point substitution process, each site is drawn from a categorical mixture of substitution processes.

$$\text{SiteMix}(u, Q_c, \pi_c) \equiv \text{Mix}_{c\sim u}\left(\text{Subst}\left(Q_c, \pi_c\right)\right)$$

where $\text{Mix}_{k\sim p}(F(\theta_k))$ denotes a mixture of component models $F(\theta_k)$ with component weights p_k and categorical class label k. Here, $c \sim u$ denotes site class.

By model design, the ancestral and descendant residues within an alignment column must share the same categorical label; site classes remain fixed and do not evolve. In order to evaluate alignment likelihoods, the unobserved latent class labels must be marginalized out. Conditioning on the ancestor sequence can make this marginalization cumbersome, so we instead evaluate the jointly-normalized likelihood $\mathrm{P}\left(A, \mathcal{X}(A) \mid t\right)$. The autoregressive decomposition uses the *next-column joint probability* $\mathcal{T}^{XY}_{\texttt{HMM}}\left(b \mid a, X_{i+1}, t\right)$. This likelihood has the same factorization as $\mathcal{T}^{Y|X}_{\texttt{HMM}}\left(b \mid a, X_{i+1}, t\right)$, but the transition and emission probabilities include additional terms to account for the ancestral sequence.

3. Proposed models

3.1. TKF-based mixtures of processes

3.1.1. Mixture of fragment classes

Instead of drawing from one fragment process, each TKF92 fragment is drawn from a categorical distribution of fragment processes. Each component fragment process also contains its own mixture over point substitution processes.

$$\begin{aligned}&\text{FragMix}(w, r_f, \theta_{cf}) \equiv \\ &\quad \text{Mix}_{f\sim w}(\text{Frag}(\text{SiteMix}(u_f, Q_{cf}, \pi_{cf}); r_f))\end{aligned}$$

where $f \sim w$ is the fragment class label and $\theta_{cf} = (u_f, Q_{cf}, \pi_{cf})$ is the collection of parameters for the inner-level substitution processes. When evaluating the joint likelihood, latent site and fragment class labels appear as state indices in an expanded Pair HMM and can be marginalized out with the forward algorithm.

3.1.2. Mixture of domain classes

Finally, we nest a mixture of TKF92-based models *inside* a TKF91 links model.

Outer model: The outer model is a TKF91 birth-death process where outer-level mortal links are born with insertion rate λ_0 and deletion rate μ_0.

Inner model: Each outer-level mortal link is associated with a subsequence of multi-residue fragments, which is generated by an inner-level TKF92 model. First, a TKF92 model is sampled from a categorical mixture; let $n \sim v$ be its domain class label. Next, an inner-level mortal link is either inserted with rate λ_n or deleted with rate μ_n. The inner-level fragment length is then drawn from a mixture of fragment processes, and each site in the fragment is drawn from a mixture of substitution models

$$\begin{aligned}&\text{DomMix}(v, \lambda_0, \mu_0, \phi_{cfn}) \equiv \\ &\text{Links}(\text{Mix}_{n\sim v}(\text{FragMix}(w_n, r_{fn}, \theta_{cfn}); \lambda_n, \mu_n); \lambda_0, \mu_0)\end{aligned}$$

where $\phi_{cfn} = (w_n, r_{fn}, \theta_{c,f,n}, \lambda_n, \mu_n)$ are the parameters for all lower-level mixture models. All latent classes manifest as additional states in the Pair HMM and are marginalized out of the joint likelihood with the forward algorithm.

3.2. Basic neural model

All previously described methods define an instantaneous evolutionary model and then attempt to solve the model's finite-time transition probabilities (exactly or approximately). As an alternative, the *Basic neural model* generates the aligned descendant sequence autoregressively, with the ancestral sequence and lapsed evolutionary time supplied as input features. We hypothesize that this model provides a reasonable approximation to the corresponding finite-time transition probabilities, despite the fact that this model is not based on a continuous-time Markov chain. Neural models do not contain latent class labels, so they are trained to maximize the conditional likelihood $\mathrm{P}\left(A \mid \mathcal{X}(A), t\right)$.

3.2.1. Concise alphabet of alignment columns.

There exist $(|\Omega|+1)^2$ possible alignment columns. However, only a subset of columns can have finite probability after conditioning on an ancestral sequence. Let $\Omega_{\texttt{aug}} = (\Omega \times \{\texttt{M}, \texttt{I}\}) \cup \{\epsilon, \texttt{E}\}$ be the concise alphabet that represents the possible columns after specifying an ancestral token. Let $\mathcal{U}(x)$ be the $|\mathcal{A} \cup \{\texttt{E}\}| \times |\Omega_{\texttt{aug}}|$ matrix that maps probabilities over the concise column alphabet to probabilities over the full column alphabet, conditioned on the ancestral token.

3.2.2. ALIGNMENT LIKELIHOOD.

In the Basic neural model, the next-column conditional probability $\mathcal{T}^{Y|X}_{\texttt{neu}}(b \mid a, i, j, X, Y_{\ldots j}, t)$ depends on position-specific embeddings from two neural networks:

- $B : \Omega^{L_X} \to \mathbb{R}^{L_X \times H}$ creates embeddings for an ancestral sequence of length L_X, with embedding dimension H. Each embedding vector $B(X)_i$ can be influenced by the entire ancestral sequence X.
- $D : \Omega^{L_Y} \to \mathbb{R}^{L_Y \times H}$ creates embeddings for a descendant sequence of length L_Y. This embedding has causal context with respect to the descendant sequence: $D(Y)_j \equiv D(Y \otimes m_{\leq j})_j$ where $m_{\leq j}$ masks all positions after j.

The next-column conditional probability is generated by

$$\begin{aligned} z_{ij} &= F\left(B(X)_{i+1}, D(Y)_j, \texttt{onehot}(\tau(a)), [t] \right) \\ \mathcal{T}^{Y|X}_{\texttt{neu}}&(b \mid a, i, j, X, Y_{\ldots j}, t) = \\ &\left(\mathcal{U}(X_{i+1}) \cdot \texttt{softmax}\left(z_{ij}\right) \right)_b \end{aligned}$$

where $\texttt{layerNorm}(\ldots)$ denotes layer normalization (Ba et al., 2016). Function F transforms the input features into logits z_{ij}. Specifically, sequence embeddings at $(i+1, j)$ are concatenated, layer normalized, and further concatenated with the previous column's one-hot encoded state type and a single-element vector containing the lapsed evolutionary time. These column-wise embeddings are passed into a two-layer feedforward network with dropout, and the final logits are transformed into probabilities over the concise alphabet using softmax.

3.2.3. SEQUENCE EMBEDDING MODELS

We evaluate three architectures for sequence embedding models B and D: a residual CNN (He et al., 2016; LeCun et al., 1989), an LSTM (Hochreiter & Schmidhuber, 1997) and a Transformer with rotary positional embeddings (RoPE) (Vaswani et al., 2017; Su et al., 2024). The residual CNN consists of a pre-norm residual block with a convolution layer, layer normalization, SiLU activation, and dropout (Ba et al., 2016; Elfwing et al., 2017; Srivastava et al., 2014). The Transformer follows a standard pre-norm architecture with a self-attention block and a two-layer multi-layer perceptron (MLP), each wrapped in residual connections, with layer normalization and SiLU activation.

3.3. Neural TKF model: a hybrid approach

We now combine instantaneous evolution processes with neural sequence embeddings. The *Neural TKF model* assumes a distinct TKF92+F81 model at every alignment column. The evolutionary model parameters are generated by neural networks from the available alignment-Markovian context. First, logits are generated by

$$\begin{aligned} (z^{\lambda}_{ij}, z^{\mu}_{ij}, z^{r}_{ij}) &= G\left(B(X_{i+1}), D(Y_j), \texttt{onehot}(\tau(a)), [t]\right) \\ z^{\pi}_{ij} &= W\left(B(X_{i+1}), D(Y_j), [t]\right) \end{aligned}$$

where G and W each transform their input features into logits z^{θ}_{ij}, with $\theta = (\lambda, \mu, r, \pi)$ being parameters of a TKF92+F81 model. Like before, sequence embeddings at $(i+1, j)$ are concatenated, layer normalized, and further concatenated with a single-element vector containing lapsed time. G also concatenates the one-hot encoded state type of the previous column, but W omits this context. The resulting column-specific embeddings are passed into separate two-layer feedforward networks with dropout to generate logits z^{θ}_{ij}. These logits are transformed into evolutionary model parameters by activation functions that ensure valid parameter values. These parameters yield site- and sample-specific TKF92+F81 transition and emission probabilities.

The Neural TKF next-column conditional probability is similar to that of the Pair HMMs.

$$\mathcal{T}^{Y|X}_{\texttt{NTKF}}(b \mid a, i, j, X, Y_{\ldots j}, t) = \mathcal{M}(b, X_{i+1}) T^{ij}_{\tau(a)\tau(b)} E^{ij}(b)$$

with a unique transition matrix T^{ij} and emission scoring function $E^{ij}(b)$ for every position (i, j).

4. Methods

4.1. Dataset curation

Pairwise alignments are curated from Pfam 36.0 (Sonnhammer et al., 1997), specifically the Pfam-A seed alignments and phylogenetic trees. First, we discard: sequences repeated within an MSA (keeping only the first instance); sequences repeated across MSAs (i.e. across Pfams); peptides shorter than 10 amino acids, sequences containing nonstandard characters (B, O, U, X, Z); and Pfams with only one member. Twelve alignments were missing corresponding trees; these are imputed with FastTree 2.1.11 (Price et al., 2010), the same program used by Pfam (Finn et al., 2014).

We extract pairwise alignments from MSAs following the method described in (Prillo et al., 2023). Briefly, the two closest leaves in the tree are pruned and paired together. This process is repeated until the entire MSA is exhausted. If a single sequence remains unpaired, it is removed from the dataset. Sequence pairs are not realigned. Although observed alignments can never be assumed to be true, our models treat alignment as a latent variable that could be marginalized out. Since we assume time-reversible models, each pairwise alignment yields two samples: one with the first sequence as the ancestor and the second as the descendant, and another sample with the roles reversed. After

postprocessing, less than 2% of pairwise alignments exceed 512 columns, and those that do have exceptionally many columns (up to 2,500). These longer alignments are removed to enable faster model prototyping.

In summary, we retained 19,909 Pfams and extracted 600,782 pairs of sequences to obtain a final training dataset of 1,201,564 pairwise alignments.

4.2. Train-dev-test partitions

Pfam clans and the remainder of families without clan labels are randomly assigned to one of ten splits. Once assigned, all the pairwise alignments of the given clan or family must be placed in the same split. This prevents homology information from unintentionally leaking to other splits. Assignment is balanced such that each split contains an approximately equal number of pairwise alignments. From these splits, three different permutations of train, dev, and test sets are created. Each partition has a 70:10:20 train-dev-test ratio.

4.3. Experiments

The optimization objective is to minimize the average of the conditional negative log-likelihood (NLL) over all alignments in training set $\mathcal{D}$:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{A \in \mathcal{D}} \text{logP}\left(A \mid \mathcal{X}(A), t\right)$$

Training is repeated across three distinct train-dev-test partitions. Models are evaluated using the total NLL on the held-out test set, averaged over the three partitions. We also measure the exponentiated cross-entropy (ECE), which is computed by (i) normalizing each alignment's NLL by the length of its descendant sequence, (ii) averaging these length-normalized values across all samples, and (iii) exponentiating the resulting average. This metric is essentially a per-column perplexity.

Our first investigation evaluates indel models that approximate or are special cases of the GGI process: TKF91, TKF92, LG05, RS07, and H20 (Thorne et al., 1991; 1992; Löytynoja & Goldman, 2005; Redelings & Suchard, 2007; Holmes, 2020). The equilibrium distribution is set to the observed amino acid frequencies, and the substitution probabilities are taken from the GTR-LG08 model. Branch lengths are provided by Pfam trees. We also briefly explored treating time as a latent variable and marginalizing over a grid of proposed times. This yielded similar likelihoods and conclusions at greater computational cost.

Moving forward, we use the observed amino acid frequencies as the equilibrium distribution only if there is one substitution process. Otherwise, distributions are fit during training. The F81 substitution model is used for emission probabilities. Our second investigation evaluates hierarchical mixtures of site, fragment, and domain classes and compares their Akaike information criterion (AIC) (Akaike, 1974) gains relative to the reference TKF92+F81 model. The number of mixture components is held constant across the lower hierarchical levels. For example, a model with two domain classes also includes two fragment classes and two site classes, yielding $2^3 = 8$ distinct combinations of labels. The numbers of mixture components tested are: 2-5, 10, 20, 30, 175, 500, and 900 for the mixture of site classes; 2-5, 10, 20, and 30 for the mixture of fragment classes; and 2-5 and 10 for the mixture of domain classes.

Our final investigation compares neural methods against the best mixture-based models. Initial trials tested six configurations of architectures formed by combining one of three sequence embedding architectures (Residual CNN, LSTM, and Transformer) with one of two prediction heads (Basic neural and Neural TKF). For each configuration, ancestor and descendant embedders share identical architectures and hyperparameters, except where necessary to enforce causal descendant sequence context.

We optimize a Transformer-based embedding architecture through hyperparameter sweeps while keeping the Neural TKF prediction head fixed. The sweeps vary the embedding dimension, number of Transformer blocks, number of self-attention heads, dropout rate, global learning rate, and weight decay. Performance is evaluated using the total NLL on a single dev set. The final Transformer-based model is called the "6-block" model, which reflects the optimal number of blocks found during the sweep. The initial Transformer configuration is called the "1-block" model. The same sequence embedding architecture is used for both the Neural TKF and Basic neural models, although only the former is explicitly optimized. We briefly explored alternative architectures for the Neural TKF prediction head (with the sequence embedders fixed), but the choice of sequence embedding architecture had more effect on model fit.

All models are trained using mini-batch stochastic gradient descent with the Adam optimizer (Kingma & Ba, 2017) on NVIDIA GeForce RTX 2080 Ti and RTX A6000 GPUs.

5. Results

5.1. Evaluation of approximations to the GGI process

Table 1 compares basic indel models and reports the total test set NLL, averaged over three train-dev-test partitions. Because TKF91 overestimates the number of indel events, it infers inflated indel rates and yields the poorest likelihoods. LG05 and RS07 make similar approximations to the finite-time transition probabilities. They have comparable performance and intermediate placement in the evaluation table, matching results from simulation studies. TKF92 demonstrates a slightly better fit to real alignments than

Table 1. **Comparing indel models by NLL of the held-out test set (lower NLL and ECE is better):** "Total NLL ($\times 10^6$)" is the NLL of test set, summed over samples. "Average NLL" is the NLL averaged over samples. The best model, TKF92, is **bolded**.

INDEL MODEL	TOTAL NLL ($\times 10^6$)	AVERAGE NLL	ECE
TKF92	**71.58**	**280.240**	**6.1763**
H20	71.64	280.461	6.1875
LG05	74.08	290.002	6.5310
RS07	74.09	290.032	6.5321
TKF91	75.70	296.362	6.7533

H20, despite H20 proving better for simulated gap profiles.

These initial investigations informed our subsequent model design and evaluations. Treating time as a latent variable and marginalizing it out increased computational cost with negligible affects to likelihood or indel model rankings. Therefore, we proceeded to evaluate likelihoods using Pfam branch lengths. Since TKF92 has a closed-form solution, is straightforward to elaborate further, and fits observed alignments best, it serves as the basis for our subsequent attempts to describe indel evolution more realistically. We accept the latent fragment boundaries inherent to TKF92 as a minor technical inconvenience that does not hurt performance.

5.2. Evaluation of TKF-based hierarchical mixtures

In Figure 1, the gain in AIC (Akaike, 1974) is plotted for all hierarchical mixture models (compared to TKF92+F81). This figure is generated from one train-dev-test partition, but trends are consistent across all three. Results demonstrate that increasing the number of site class components improves model fit, as compared to assuming only one substitution process; this is consistent with prior work (Yang, 1994; Le et al., 2008). However, across all latent class types, increasing the number of components yields diminishing improvements in model fit. In particular, as the number of components grows, models exhibit component collapse and redundancy. Better fit is achieved only by designing higher levels of process, as in our DomMix and FragMix models.

This behavior aligns with our expectations about how heterogeneous evolutionary forces are structured. Coordinating substitution mixtures within each fragment class reflects the expectation that residues collocated in sequence (and hence in space) are often subject to similar substitution pressures. Synchronizing mixtures of fragment processes (along with fragment-level indel rates) within each domain class reflects the fact that larger structural or functional regions have specific tolerances to indel events, in addition to known substitution biases. For example, the hydrophobic core of a globular protein tends to have poor tolerance for big insertions or hydrophilic amino acids.

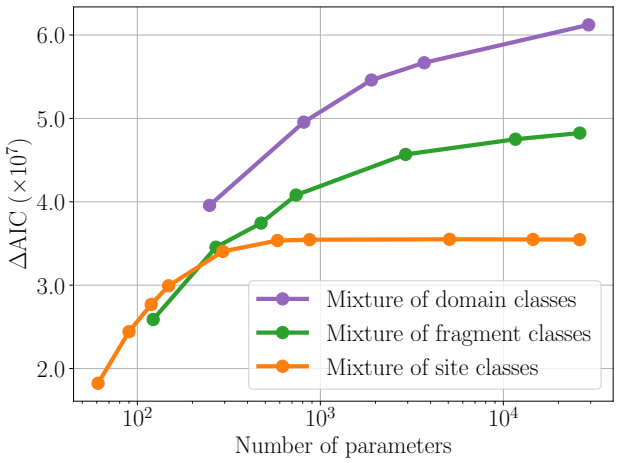

Figure 1. **TKF-based mixtures of processes: gain in AIC over TKF92+F81 (higher is better):** Each dot corresponds to a single trial with a fixed number of mixture components, detailed in Section 4.3. The x-axis shows the log-scaled number of model parameters and the y-axis shows gain in AIC over the reference model, TKF92+F81. Each curve asymptotically approaches (or is projected to approach) a limit on the performance gains obtained from increasing the number of components.

Practically, optimizing a hierarchical mixture model requires few design choices. A relatively small number of parameters means no model ever overfits the training data. Although substantially larger mixtures would likely overfit, increasing to such scale would be unjustifiable due to diminishing returns in performance.

5.3. Evaluation of neural and mixture-based models

Table 2 compares all neural models against the best-performing hierarchical mixtures and reports the total test set NLL, averaged over three train-dev-test partitions. The overall best-performing model is the Neural TKF with 6-block Transformer sequence embedders, followed by the same prediction head with LSTM embedders. However, by total NLL, the 10-component mixture of domain classes ranks third overall, despite using roughly three orders of magnitude fewer parameters than the neural methods. The difference in NLL between this mixture-based model and all neural models is smaller than the gap between LG05 and H20 (74.08 vs. 71.64×10^6), which are trained on the exact same partitions and evaluated with identical branch lengths. The 30-component mixture of fragment classes also proves moderately comparable, as it outperforms the Basic neural model with 1-block Transformer sequence embedders.

When evaluated using ECE, all but two neural models achieve better fit than the 10-component mixture of domains model. That is, neural methods are, on average, more confi-

Table 2. **Comparing neural models against TKF-based hierarchical mixture models by NLL of held-out test set (lower NLL and ECE are better):** "Number of parameters" is the number of trainable parameters in the model. Other column headings match those in Table 1. The best model, Neural TFK with 6-block Transformer sequence embedders, is **bolded**. We also mark the 10-component mixture of domain classes with a star ($\star$) for its parameter efficiency and for achieving a lower NLL than all but two neural models.

MODEL TYPE	MODEL SUBTYPE	NUMBER OF PARAMETERS	TOTAL NLL ($\times 10^6$)	AVERAGE NLL	ECE
NEURAL TKF (SEC. 3.3)	**6-BLOCK TRANSFORMER**	**43.55 M**	**61.72**	**246.867**	**5.5169**
NEURAL TKF	LSTM	30.08 M	61.85	247.388	5.5340
NEURAL TKF	RESIDUAL CNN	28.53 M	62.47	249.864	5.6258
NEURAL TKF	1-BLOCK TRANSFORMER	29.78 M	62.99	251.962	5.6983
BASIC NEURAL (SEC. 3.2)	LSTM	27.00 M	62.77	251.068	5.6877
BASIC NEURAL	6-BLOCK TRANSFORMER	42.04 M	62.82	251.282	5.6992
BASIC NEURAL	RESIDUAL CNN	26.47 M	63.17	252.661	5.7455
BASIC NEURAL	1-BLOCK TRANSFORMER	26.87 M	64.04	256.156	5.8817
MIXTURE OF					
...DOMAIN CLASSES (SEC. 3.1.2)	10 COMPS.	$\star$ 29.23 K	$\star$ 62.14	$\star$ 248.541	$\star$ 5.7013
...FRAGMENT CLASSES (SEC. 3.1.1)	30 COMPS. ($k_n = 1$)	26.16 K	63.87	255.479	5.9602
...SITE CLASSES (SEC. 2.6)	30 COMPS. ($k_n = k_f = 1$)	873	65.00	260.028	6.1238
TKF92+F81 (SEC. 2.4)	N/A	11	70.30	281.203	7.1398
TKF91+F81 (SEC. 2.4)	N/A	10	73.87	295.506	7.7942

dent in their per-column predictions, whereas the mixture of domains model better captures the total likelihood. The discrepancy in model rankings suggests that performance may depend on underlying factors such as descendant length.

Finally, for any given sequence embedding architecture, the Neural TKF prediction head always achieved better model fit than the Basic neural prediction head; this is observed when calculating both total NLL and ECE. These results suggest that incorporating the inductive bias of evolutionary modeling improves neural network performance.

6. Discussion

In this work, we curate a training dataset of pairwise alignments and branch lengths from Pfam, and we re-evaluate indel models that approximate or reduce to the GGI indel process. (Holmes, 2020) demonstrated that H20 was better than TKF92 at recovering gap profile distributions, as simulated by the GGI process. However, our results show that TKF92, not H20, is a better fit for real data. This is surprising but not unexpected; simulated gap profiles are known to exhibit different length distributions than empirical ones (Wygoda et al., 2024).

We extend TKF92 into increasingly nested hierarchical mixture models and design two classes of neural models. Overall, the hierarchical mixtures were far easier to optimize than the neural methods. Our neural hyperparameter sweeps explored only a small fraction of the possible architectures. Additionally, the primary challenge during hyperparameter sweeps was scaling up without inducing overfitting to the training set. In contrast, the hierarchical mixtures had fewer design choices and never overfit.

During model evaluation, adding hierarchical levels gave a greater boost to model fit than increasing the number of components in any single level. Overall, hierarchical mixtures are competitive with neural methods at fitting Pfam alignments, despite having three orders of magnitude fewer parameters. Selection is heterogeneous across residues and context-dependent due to biophysical constraints. Neural models incorporate complex inter-residue interactions by design. Our results suggest that adding hierarchical mixtures to evolutionary models allows them to capture progressively more complex and realistic selection forces. In particular, the mixture of domains model introduces latent domain classes that contain coordinated fragment-level indel rates and substitution processes. Thus, surrounding sequence context can indirectly influence indel dynamics, despite the two being independent processes. This may explain why the performance of the mixture of domains model is comparable to much larger neural networks.

Finally, neural models with the inductive bias for evolutionary models consistently outperform their counterparts without bias. Similar conclusions have been drawn in several prior studies. (Prillo et al., 2024) found that fitting a point substitution model to every alignment column produced a variant effect prediction model that outperformed neural methods. (Hsu et al., 2022) also found that an augmented Potts model (which is suggested to capture evolutionary signals) was competitive with the neural protein language models of the time at predicting function from sequence.

In conclusion, CTMC-based models remain a relevant framework for describing molecular evolution, even in the age of large neural networks.

Acknowledgments

We thank Yun Song, Sebastian Prillo, and Antoine Koehl for helpful discussions and feedback. This work was funded by NIH grants HG004483 and GM080203. GPUs were provided in part by the NVIDIA Academic Hardware Grant.

References

Akaike, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, December 1974. ISSN 1558-2523. doi: 10.1109/TAC.1974.1100705. URL https://ieeexplore.ieee.org/document/1100705.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer Normalization, July 2016. URL http://arxiv.org/abs/1607.06450. arXiv:1607.06450 [stat].

De Maio, N. The Cumulative Indel Model: Fast and Accurate Statistical Evolutionary Alignment. *Systematic Biology*, 70(2):236–257, March 2021. ISSN 1063-5157. doi: 10.1093/sysbio/syaa050. URL https://doi.org/10.1093/sysbio/syaa050.

Durbin, R., Eddy, S. R., Krogh, A., and Mitchison, G. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, 1998. ISBN 978-0-521-62971-3. doi: 10.1017/CBO9780511790492. URL https://www.cambridge.org/core/books/biological-sequence-analysis/921BB7B78B745198829EF96BC7E0F29D.

Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning, November 2017. URL http://arxiv.org/abs/1702.03118. arXiv:1702.03118 [cs].

Felsenstein, J. Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, November 1981. ISSN 1432-1432. doi: 10.1007/BF01734359. URL https://doi.org/10.1007/BF01734359.

Finn, R. D., Bateman, A., Clements, J., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Heger, A., Hetherington, K., Holm, L., Mistry, J., Sonnhammer, E. L. L., Tate, J., and Punta, M. Pfam: the protein families database. *Nucleic Acids Research*, 42(Database issue): D222–D230, January 2014. ISSN 0305-1048. doi: 10.1093/nar/gkt1223. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC3965110/.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, June 2016. doi: 10.1109/CVPR.2016.90. URL https://ieeexplore.ieee.org/document/7780459.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Holmes, I. A probabilistic model for the evolution of RNA structure. *BMC Bioinformatics*, 5(1):166, October 2004. ISSN 1471-2105. doi: 10.1186/1471-2105-5-166. URL https://doi.org/10.1186/1471-2105-5-166.

Holmes, I. A Model of Indel Evolution by Finite-State, Continuous-Time Machines. *Genetics*, 216(4): 1187–1204, December 2020. ISSN 0016-6731. doi: 10.1534/genetics.120.303630. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC7768254/.

Hsu, C., Nisonoff, H., Fannjiang, C., and Listgarten, J. Learning protein fitness models from evolutionary and assay-labeled data. *Nature Biotechnology*, 40(7):1114–1122, July 2022. ISSN 1546-1696. doi: 10.1038/s41587-021-01146-5. URL https://www.nature.com/articles/s41587-021-01146-5.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization, January 2017. URL http://arxiv.org/abs/1412.6980. arXiv:1412.6980 [cs].

Knudsen, B. and Miyamoto, M. M. Sequence Alignments and Pair Hidden Markov Models Using Evolutionary History. *Journal of Molecular Biology*, 333(2):453–460, October 2003. ISSN 0022-2836. doi: 10.1016/j.jmb.2003.08.015. URL https://www.sciencedirect.com/science/article/pii/S0022283603009987.

Lanave, C., Preparata, G., Sacone, C., and Serio, G. A new method for calculating evolutionary substitution rates. *Journal of Molecular Evolution*, 20(1):86–93, February 1984. ISSN 1432-1432. doi: 10.1007/BF02101990. URL https://doi.org/10.1007/BF02101990.

Le, S. Q. and Gascuel, O. An Improved General Amino Acid Replacement Matrix. *Molecular Biology and Evolution*, 25(7):1307–1320, July 2008. ISSN 0737-4038. doi: 10.1093/molbev/msn067. URL https://doi.org/10.1093/molbev/msn067.

Le, S. Q., Lartillot, N., and Gascuel, O. Phylogenetic mixture models for proteins. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 363 (1512):3965–3976, October 2008. ISSN 0962-8436. doi:

10.1098/rstb.2008.0180. URL https://doi.org/10.1098/rstb.2008.0180.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. Handwritten Digit Recognition with a Back-Propagation Network. In *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989.

Löytynoja, A. and Goldman, N. An algorithm for progressive multiple alignment of sequences with insertions. *Proceedings of the National Academy of Sciences*, 102(30):10557–10562, July 2005. doi: 10.1073/pnas.0409137102. URL https://www.pnas.org/doi/full/10.1073/pnas.0409137102.

Price, M. N., Dehal, P. S., and Arkin, A. P. FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments. *PLOS ONE*, 5(3):e9490, March 2010. ISSN 1932-6203. doi: 10.1371/journal.pone.0009490. URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0009490.

Prillo, S., Deng, Y., Boyeau, P., Li, X., Chen, P.-Y., and Song, Y. S. CherryML: scalable maximum likelihood estimation of phylogenetic models. *Nature Methods*, 20(8):1232–1236, August 2023. ISSN 1548-7105. doi: 10.1038/s41592-023-01917-9. URL https://www.nature.com/articles/s41592-023-01917-9.

Prillo, S., Wu, W., and Song, Y. S. Ultrafast classical phylogenetic method beats large protein language models on variant effect prediction. *Advances in Neural Information Processing Systems*, 37:130265–130290, December 2024. doi: 10.52202/079017-4139.

Redelings, B. D. and Suchard, M. A. Incorporating indel information into phylogeny estimation for rapidly emerging pathogens. *BMC Evolutionary Biology*, 7(1):40, March 2007. ISSN 1471-2148. doi: 10.1186/1471-2148-7-40. URL https://doi.org/10.1186/1471-2148-7-40.

Sonnhammer, E. L., Eddy, S. R., and Durbin, R. Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins*, 28(3):405–420, July 1997. ISSN 0887-3585. doi: 10.1002/(sici)1097-0134(199707)28:3⟨405::aid-prot10⟩3.0.co;2-l.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014. ISSN 1532-4435.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, February 2024. ISSN 0925-2312. doi: 10.1016/j.neucom.2023.127063. URL https://www.sciencedirect.com/science/article/pii/S0925231223011864.

Thorne, J. L., Kishino, H., and Felsenstein, J. An evolutionary model for maximum likelihood alignment of DNA sequences. *Journal of Molecular Evolution*, 33(2):114–124, August 1991. ISSN 1432-1432. doi: 10.1007/BF02193625. URL https://doi.org/10.1007/BF02193625.

Thorne, J. L., Kishino, H., and Felsenstein, J. Inching toward reality: An improved likelihood model of sequence evolution. *Journal of Molecular Evolution*, 34(1):3–16, January 1992. ISSN 1432-1432. doi: 10.1007/BF00163848. URL https://doi.org/10.1007/BF00163848.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is All you Need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wilburn, G. W. and Eddy, S. R. Remote homology search with hidden Potts models. *PLOS Computational Biology*, 16(11):e1008085, November 2020. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1008085. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1008085. Publisher: Public Library of Science.

Wygoda, E., Loewenthal, G., Moshe, A., Alburquerque, M., Mayrose, I., and Pupko, T. Statistical framework to determine indel-length distribution. *Bioinformatics*, 40(2):btae043, February 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae043. URL https://doi.org/10.1093/bioinformatics/btae043.

Yang, Z. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: Approximate methods. *Journal of Molecular Evolution*, 39(3):306–314, September 1994. ISSN 1432-1432. doi: 10.1007/BF00160154. URL https://doi.org/10.1007/BF00160154.