Characterization of selective pressures acting on protein sites with Deep Learning

Estelle Bergiron[1], Luca Nesterenko[1], Julien Barnier[1], Philippe Veber[1], and Bastien Boussau[1,*]

[1]Univ Lyon, Univ Lyon 1, CNRS, VetAgro Sup, UMR5558, Laboratoire de Biométrie et Biologie Evolutive, F-69100, Villeurbanne, France

July 22, 2025

Abstract

It is often useful, in the field of molecular evolution, to identify the selective pressures acting on a particular site of a protein to better understand its function. This is typically done with likelihood-based approaches applied to codon sequences in a phylogenetic context. However, these approaches are computationally costly. Here we adapt a linear transformer neural network architecture, which has been shown to be able to reconstruct accurate phylogenies from sequence alignments, to identify selective pressures acting on individual amino acid sites. We design different versions of the architecture and train and test them on simulations. We compare the results of one of our best models to the state-of-the-art approach codeml and find that it outperforms it when it is applied to data that resemble its training data, but that it performs less well when applied to data that does not resemble the training data. In all cases, our approach operates at a fraction of codeml's computational cost. These results suggest that such a neural network architecture, trained on realistic simulations, could compare very favorably to state-of-the-art approaches to characterize selection pressures acting on coding sequences.

1 Introduction

Deep learning approaches have been very successful in a range of scientific areas [15, 5, 1]. In recent years, they have begun to be applied in genomics and phylogenetics [7]. They could be very useful for predicting site-wise features in protein-coding genes, such as substitution rates. In support of this idea, [16] have reported highly accurate substitution rates inferred from sequence alignments using a deep network. Beyond substitution rates, networks could also be

used to predict the selective pressures acting on individual sites. Such inference is usually based on the d_N/d_S ratio between synonymous and non-synonymous codon substitution rates. When at a site $d_N/d_S < 1$, the ratio is indicative of negative or purifying selection; when $d_N/d_S \approx 1$, of neutral evolution; when $d_N/d_S > 1$, of positive selection. Methods that estimate site-wise d_N/d_S are computationally expensive, and could be advantageously replaced by a deep-learning approach. Recently the Phyloformer architecture was shown to be able to provide accurate estimates of the phylogenetic history of a sequence alignment[9, 11]. Phyloformer is based on several transformer blocks, implementing a custom linear attention mechanism that takes into consideration correlations between sites and between sequences to yield the network's estimates. Although it has been used so far to provide estimates of distances between sequences, we reasoned that it could also provide site-wise estimates of selective regimes.

In this paper, we present our efforts to build upon the Phyloformer architecture to classify protein sites as under neutral evolution or negative or positive selection. We train our network, SelRegAA, on simulations of protein alignments and investigate its performance as we modify its structure and size. We also investigate its sensitivity to the parameters of the simulation, and its statistical calibration. We show on independent simulations that it performs with an accuracy rivalling the state-of-the-art d_N/d_S method codeml, while being 3 orders of magnitude faster. However, the performance of SelRegAA decreases if it is applied to data that does not look like the data it has been trained on. Overall, we conclude that a linear transformer network, operating on codon sequences, and trained on data representative of the range of alignments observed empirically, could perform very well, at a fraction of the cost of codeml.

2 Methods

2.1 Neural network architecture

Our neural network is based on Phyloformer [10], with two main differences (fig. 1). First, we changed the output of the network. Phyloformer outputs a distance matrix between sequences whereas in this setting we need site-wise annotations. Therefore SelRegAA outputs a vector of 3 class probabilities per site of the alignment. The loss is computed as the cross-entropy between the output class probabilities and the true site-wise selection regime. Second, SelRegAA performs attention between sequences whereas Phyloformer performs attention between pairs of sequences.

The input to the network is a one-hot encoding of an amino acid sequence alignment. Each amino acid in each sequence is thus transformed into a vector of size 21, corresponding to 20 amino acids, plus a gap character. These vectors are then converted into vectors of size 32, corresponding to the embedding dimension. The value of these embedding vectors is updated as they traverse the network and its 6 attention blocks. At the end, the embedding vectors are converted into vectors of size 3, which are used as predictors for neutral

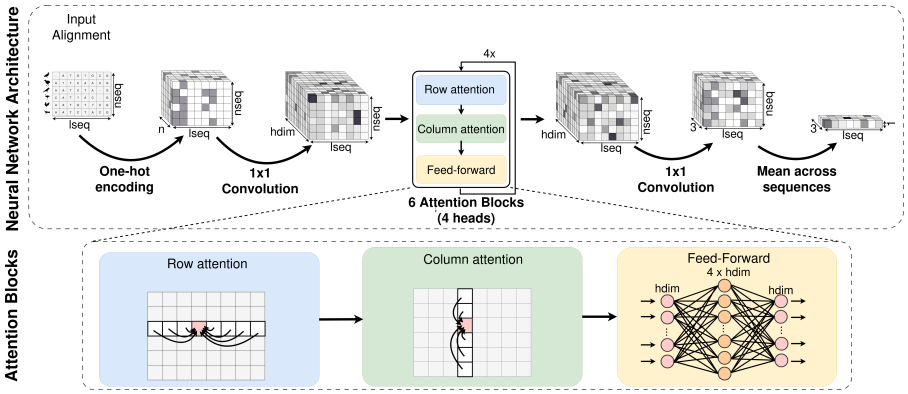


Figure 1: Structure of the base network. Top row: general architecture, from amino acid sequence encoding (left) to site-wise prediction of selective regimes (right). Bottom row: description of an attention block with 4 heads.

evolution, negative selection, and persistent positive selection.

2.2 Training algorithm

We trained our neural network with the AdamW optimizer with a learning rate of 0.001. We also used a scheduler with a 5 epochs warmup period followed by an exponential decay with a parameter γ of 0.99. The network was trained for a maximum of 200 epochs with an early stopping criterion, halting training if the validation loss value didn't go down for 5 consecutive epochs.

The network was trained with 10000 alignments, 90% of which for training, and 10% for validation.

2.3 Simulation protocol

Alignments were simulated as in [10]. Briefly, ultrametric tree topologies were simulated using a Birth-Death process. Branch lengths were then altered to introduce heterogeneities between root-to-tip distances, and the trees were rescaled to match empirical tree diameters observed in the HOGENOM database [12] and the RaxML-Grove database [3]. Then, alignments were simulated along the phylogenies using the sequence simulator Pastek. For each site, Pastek randomly drew a selective regime category. For each alignment, the proportions of sites under neutral evolution and under positive selection are both drawn from a Poisson distribution of rate 5 while the remaining sites are simulated under negative selection. All sites were simulated according to the Mutation-Selection model [14], with the effective population size parameter drawn from a uniform distribution between 0.5 and 5. Neutral sites were simulated assuming a flat amino acid profile, where all amino acids have the same fitness. Negatively selected sites were simulated with an amino acid profile drawn from 263 empir-

ically obtained profiles [2]. Positively selected sites were simulated by drawing an amino acid profile as for negatively selected sites, but with an additional parameter to model persistent escape from the current amino acid [17]. Gaps were simulated in the alignments with a deletion rate drawn from $0.05, 0.1, 0.5$ with weights $0.15, 0.7, 0.15$, and according to the TKF91 model [18]. An insertion rate equal to 0 was chosen to avoid the burden of having to choose a selective regime for sites appearing later than at the root. The resulting alignments were 250 sites long, and contained 50 sequences.

2.4 Performance metrics

To evaluate the performance of our networks, we report precisions, recalls and F_1 scores. The precision for a class is the proportion of correctly predicted sites among all the sites predicted by the network to belong to that class. The recall for a class is the proportion of correctly predicted sites among all the sites in the dataset belonging to that class. The F_1 score for a class is the harmonic mean of the precision and recall scores for that class. It ranges between 0 and 1.0, with 1.0 for a perfect prediction with a recall and a precision of 1.0. As a global performance metric we use a global F_1 score as an average of the F_1 scores of each class.

3 Results

3.1 Network architecture and training

In this section we compare the performances of our base model to variants of it, where the training or the architecture of the network were changed.

3.1.1 Training parameters

Our base model was trained on 10000 alignments. We performed 4 independent trainings of our base network architecture to evaluate the reproducibility of our training. Supplementary Fig. 1 shows that all replicate trainings resulted in very similar performances, with F_1 scores ranging between 0.765 (model base 2) and 0.774 (model base 1). Models 3 and 4 obtained F_1 scores of 0.770. Due to our stopping criterion, training times varied, with model base 1's training taking 1h43 minutes for 69 epochs, model base 2's training taking 54 minutes for 38 epochs, model base 3 1h10 for 45 epochs, and model base 4 1h47 for 49 epochs. This experiment suggests that differences of around 0.01 in F_1 scores can be observed between replicates. In terms of precision and recall, the largest differences are observed for classifying sites under positive selection, reaching around 0.04.

We explored the network's performance when it is trained on a larger number of sequences, i.e. 50000 alignments instead of 10000. After training on the larger dataset, the network obtains an F_1 score of 0.793, better than the 0.774 of base model 1. In particular, both precision and recall are improved for detecting

positively selected sites with the larger training dataset size (Supp. Fig. 2). In the following tests, we train the variants of our network on 10000 alignments to avoid large computational footprints.

3.1.2 Varying the network architecture

Our base network uses attention between sequences and thus differs from Phyloformer, which works on pairs of sequences to provide accurate estimates of pairwise distances. Attempts at working at the sequence level to reconstruct pairwise distances had been much less accurate [9]. However, working on pairs of sequences entails a memory footprint that scales quadratically with the number of sequences in the alignment, compared to linearly when working with sequences. We investigated the performance of an architecture operating on pairs of sequences, and compared it to the base model, which works on individual sequences.

Fig. 2 shows that the pair model performs better than the sequence model, in particular for the detection of positive selection, with a gain in precision of 0.08, and in recall of 0.06. These gains appear significant since variation between our four replicates (Sup. Fig. 1) were within a range of 0.04. They likely result from the gain in expressivity associated with considering pairs instead of individual sequences, and not from the number of parameters, because the sequence model has $102,051$ parameters, and the pair model has $68,323$ parameters. The pair model has fewer parameters because only 4 attention blocks were used instead of 6, to allow a batch size of 8 alignments during training. However, training the pair model was 15 times slower than the sequence model, and had a markedly larger memory footprint. In the following, we choose to focus on the sequence model because it has the best potential for scaling up to large datasets, where fast approaches are most needed.

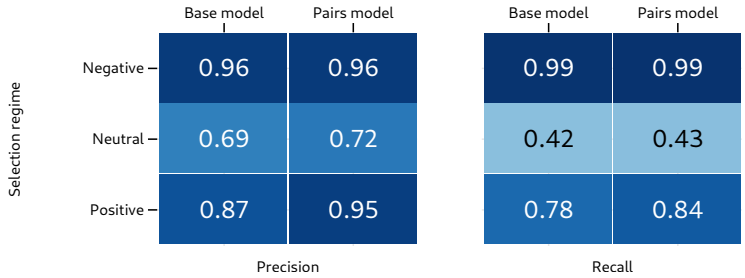


Figure 2: A sequence level model performs less well than a costly pair-level model.

Size of the embedding The base network takes a sequence alignment as input, in which all sites are one-hot encoded into vectors of dimension 21. In turn, these vectors are embedded into vectors of size 32, that then pass through the attention blocks. We investigated the impact of changing the size of the

embedding. Supplementary Fig. 3 shows that the size of the embedding has little influence on the performance of the network. Decreasing it to 16 instead of 32 results in a slightly lower F_1 score of 0.755, and increasing it to 48 results in the same F_1 score of 0.774.

Number of attention blocks and heads We investigated a range of network sizes, by changing the number of attention blocks, attention heads and embedding size in our network.

Our base model has 6 attention blocks that each combine column and row attention mechanisms. We tested reducing this number to 2 and 4, or increasing it to 8. Supplementary Fig. 4 shows that the performance of the network is not very sensitive, if at all, to the number of attention blocks.

Inside each axial attention mechanism, we use several heads whose roles are to focus on different characteristics of the data. Our base model has 4 attention heads. We trained alternative models with 2 or 6 attention heads, keeping head size constant (size 8). As a result, embeddings become larger with more heads: the network with 2 heads uses embeddings of size 16, and the network with 6 heads uses embeddings of size 48. Supplementary Fig. 5 shows that the number of heads affects the performance of the network. We observe a very weak decrease in performance when only 2 heads are used (F_1 score of 0.757 instead of 0.774 for base model 1), but no improvement when 6 heads are used (F_1 score of 0.776).

The effects of row and column attentions The Phyloformer architecture performs both row and column attention in each attention block. To investigate the impact of each type of attention on site-wise (i.e., column-wise) classification accuracy, we trained two networks in which either attention had been deactivated and one network in which both attention mechanisms had been deactivated. Fig. 3 shows that both attention mechanisms are important. However, attention across sequences (across columns) is the most important, as disabling it completely removes the network's predictive ability. Attention across sites (across rows) is still important, as removing it degrades the F_1 score from 0.744 for base model 1 to 0.538. Just removing the attention across sequences is enough to completely destroy the network's ability to predict site categories: the network then predicts that all sites belong to the "negative selection" category, because it is the category with the largest number of sites in the data.

3.2 Sensitivity to dataset characteristics

3.2.1 Sensitivity to dataset size

Our network was trained on datasets with 50 sequences and 250 sites. We investigated its performance on alignments with larger numbers of sites and sequences. When trained and tested on alignments with 100 sequences instead of 50, we observe that the network's recall increases both for the neutral and

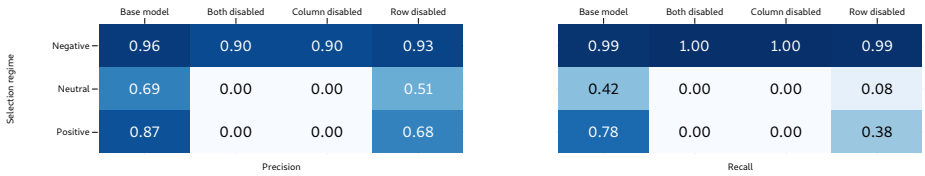


Figure 3: Effect of row and column attention on precision and recall. Compared to base model 1 (left), removing the two axial attention mechanisms degrades performance. When both attention mechanisms are disabled or when only attention across sites is used, the network fails to predict site categories (second and third columns). When only attention across sequences is used, the network's performance is decreased markedly (fourth column).

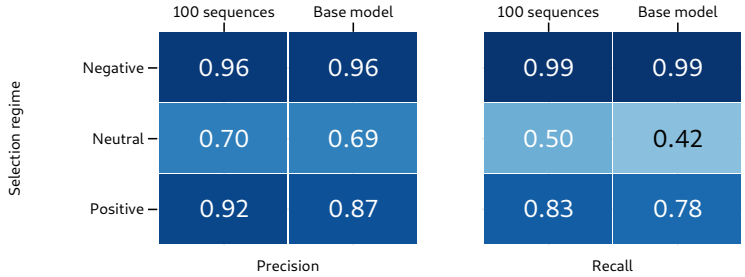


Figure 4: The trained network has better performance when it is trained and tested on alignments with a larger number of sequences.

positive selection categories, with a better precision for positive selection sites (Fig. 4).

When trained on alignments with 400 sites instead of 250, the performance of the network does not improve (Supp. Fig. 6). A larger number of sites does not result in an improved prediction accuracy per site. Increasing the number of sites in the alignment can only be useful for more accurately accounting for the phylogenetic structure in the data, which does not seem to be useful here. Conversely, since the prediction is made at the level of individual sites, a larger number of sequences provides more information per site, resulting in a higher performance. Interestingly, the higher performance obtained on alignments with 100 sequences does not translate to smaller alignments: the network trained on large alignments performs less well on alignments with 50 sequences than the network trained on smaller alignments (data not shown).

3.2.2 Weighting the loss to account for imbalance in the proportions of the categories

The training data was generated with around 5% of the sites under neutral evolution, 90% under negative selection, and 5% under persistent positive selection, on average. This strong imbalance in the proportions of the different categories

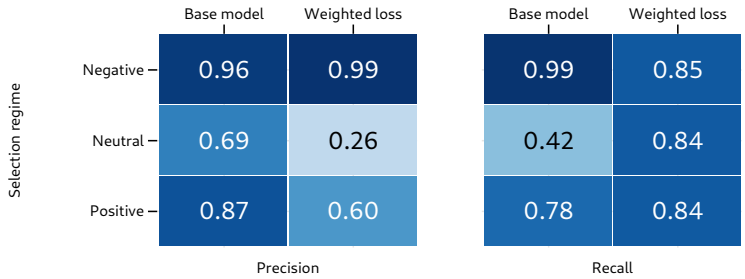


Figure 5: Precision and recall values for base model (left), and for a model trained with weighted loss (right). The model trained with weighted loss has better recall in the minor categories, and a higher precision in the most frequent category.

could lead the network to focus on accurately predicting sites under negative selection, with a poorer performance for the other two minor categories. To investigate the strength of this effect, we trained a network with a weighted loss on the same training data. This weighted loss effectively penalizes errors in the minor categories more than errors in the negative selection category, by weighting errors with the inverse of each class frequency. The weights were 0.378 for negative selection and 6.66 for the other two categories. Fig. 5 shows that the model trained with weighted loss has better recall in the minor categories and a higher precision in the most frequent category. However, the precision in the minor categories is markedly decreased. Overall, the performance of the weighted model is reduced, with an F_1 score of 0.671 compared to 0.744 for base model 1. This is not unexpected as the F_1 score depends on class balance, but in the following we focus on the unweighted models.

3.2.3 Sensitivity to the trees used

The trees in our training and testing datasets have been simulated as in [10]. Briefly, all of these trees have 50 leaves, and have been scaled to match the distribution of tree diameters observed in the HOGENOM database [12]. We investigated the sensitivity of our base model to different tree topologies. We obtained 4 empirical phylogenies used in [2]. The "besnard2009" tree is a 79 leaf phylogeny of the Cyperaceae family of flowering plants, "online_rodent" is a 32 leaf phylogeny of rodent species, "orthomam" is a 116 leaf phylogeny of mammalian species, and "rubisco" is a 179 leaf phylogeny of the Amaranthaceae family of flowering plants. Table 1 presents the F_1 scores for a variety of experiments. Performance on alignments simulated along the empirical trees (column *Empirical tree*) is systematically lower than the performance obtained on validation data during training (F_1 score of 0.774). Within the empirical trees, we observe that performance is better when the number of sequences is larger, which is consistent with our results obtained above (Fig. 4). Rescaling the empirical trees to match the tree diameters used in our training simulations

improves the performance of the network (column *Scaled empirical tree*). Performance on alignments simulated on new trees with the same number of leaves as the empirical phylogenies, but keeping the scaling as in the training data, results in similar performance (column *Simulated tree*). These results show that both the number of sequences and the scale of the tree have large impacts on the performance of the network, but the topology of the trees, whether they are simulated or empirical, does not seem to matter.

Tree	**Empirical tree**	**Scaled empirical tree**	**Simulated tree**
besnard2009	0.711	0.796	0.793
online_rodent	0.624	0.712	0.729
orthomam	0.615	0.804	0.807
rubisco	0.586	0.834	0.818

Table 1: Performance (F_1 score) of the network on different phylogenies of different lengths. The F_1 score obtained on validation data during training is 0.774.

3.3 Comparison to codeml

3.3.1 Performance on data similar to the training data

We compared our network to the state-of-the-art method codeml [19] on simulated data. codeml was used on codon sequences to predict whether a site is under negative selection, neutral evolution or positive selection (specified as model 0 in the file "codeml.ctl"). We simulated 10 alignments from 10 different trees as in the training data. Prediction of the selection regimes took about 5s on a single CPU with our base model, vs 3h on 10 CPUs for codeml. Overall, the base model has better results than codeml (Fig.6 ; Supp. Fig. 7). In particular, codeml's precision for the Neutral evolution and Positive selection classes is much lower than SelRegAA's precision. codeml's recall however is better than SelRegAA's for the Neutral evolution class, but lower for the two other classes. Overall, for a given recall rate, SelRegAA often offers a better precision than codeml (Supp. Fig. 7).

3.3.2 Calibration

We investigated whether the probabilities output by the base model and by codeml are calibrated. A probability p of an event is calibrated if the event occurs a fraction p of the time. Our network outputs for each site a probability that it is under negative, neutral, or positive selection. codeml uses Bayes Empirical Bayes estimates to compute probabilities for each site to be in one of the three selective regimes [20]. We used the results on the 10 alignments to compute the frequency with which a prediction is correct and compared it to the probabilities output by the two methods.

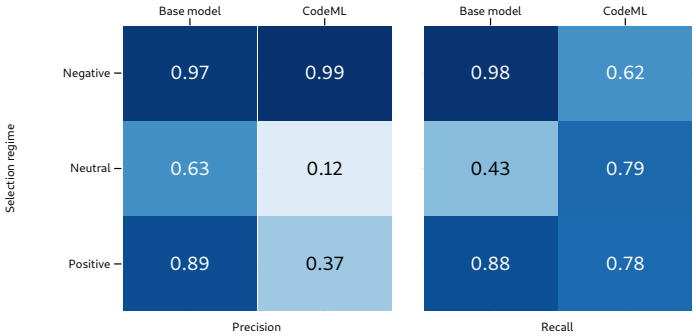


Figure 6: Precision and recall values for codeml and our basemodel on 10 alignments simulated as during training.

Fig. 7 shows that the network's output probabilities are well calibrated on data similar to its training dataset. Although the neutral and positive categories are much less frequent in the training data than the negative selection category, all three categories are predicted with good calibration. In contrast, codeml's probabilities are not well calibrated in all three categories. codeml is overly conservative for neutral and positive categories and, in reverse, overestimates the probability that a site is in the negative selection category.

3.3.3 Performance on simulations using different amino acid profiles

Our method was trained and tested on a limited set of 263 experimentally determined amino acid fitness profiles. We tested its performance on alignments simulated from independently-obtained amino acid profiles and compared it to codeml. We retrieved amino acid profiles estimated in [6] and randomly picked two samples of 250000 profiles ($profiles_0$ and $profiles_1$) from 1000 alignments.

Fig. 8a shows that the performance of the network suffers significantly when tested on alignments simulated with a different set of profiles, with an F_1 score dropping from 0.769 to about 0.575. In particular, the two classes with the smallest numbers of sites are much less well detected, both in terms of precision and recall.

A network trained on alignments simulated with $profiles_0$ has reduced performance on alignments simulated with the 263 experimentally determined amino acid fitness profiles (F_1 score of 0.676) (Supp. Fig. 8). However, performance does not vary whether test alignments have been simulated with $profiles_0$ or $profiles_1$, which both were inferred from large numbers of sites in protein alignments from mammals.

On data simulated with $profile_0$, comparison between codeml and the base model trained on the 263 default profiles shows that the performance gap between the two models has decreased with F_1 scores of 0.539 and 0.548 respectively (compared to 0.492 and 0.791 on data simulated with the 263 default profiles). The two methods, however, exhibit different behaviours (Fig. 8b).

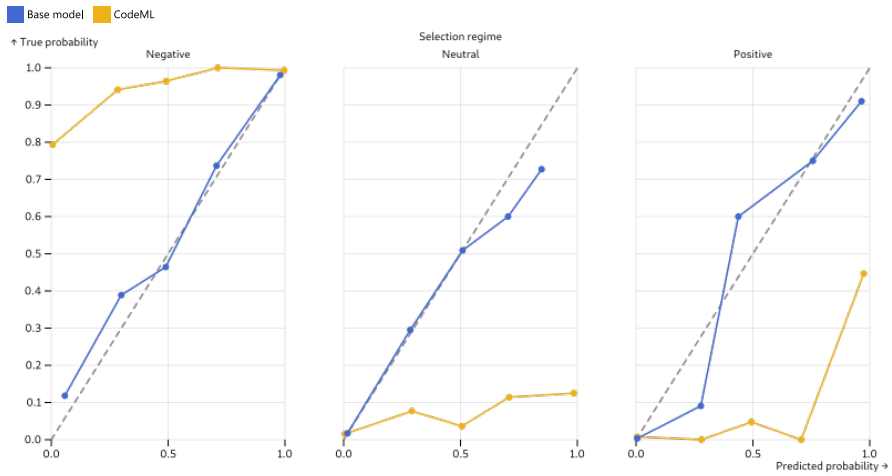
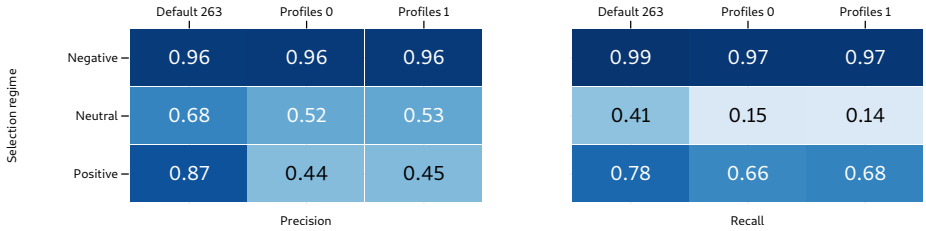
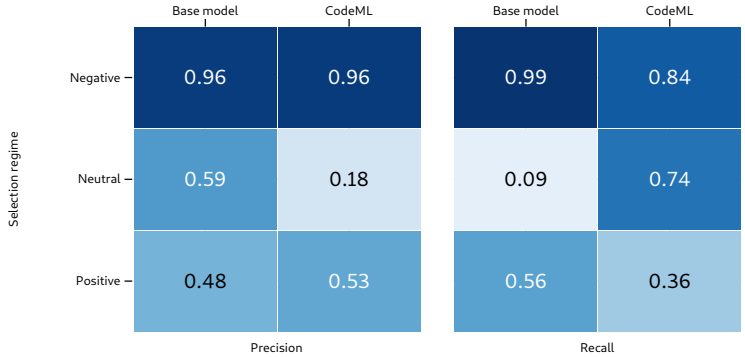


Figure 7: Calibration of the base model and of codeml.



(a) Performance of the base model is markedly decreased on alignments simulated with a different set of amino acid profiles.



(b) Comparison between codeml and the base model (trained on the default 263 profiles) on alignments simulated with $profile_0$.

Figure 8: Influence of the profiles used for training on performance

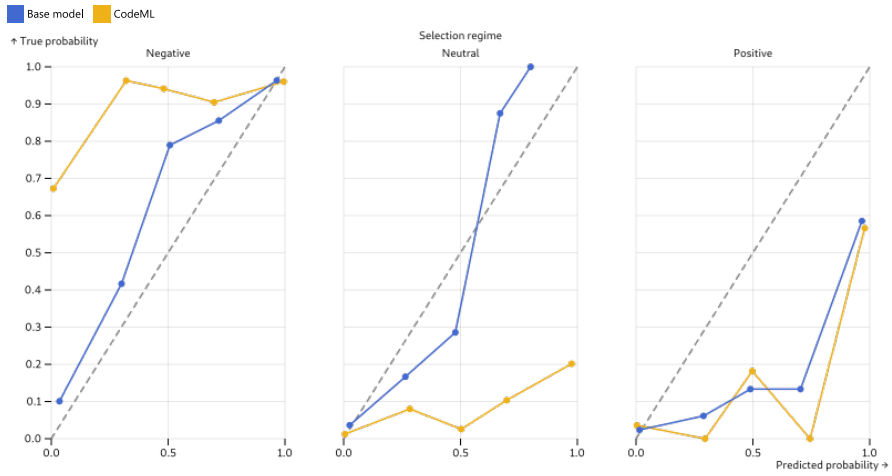


Figure 9: Calibration of codeml and the basemodel on simulations using amino acid profiles different from the training data

For example, the two methods have opposite positions on the precision-recall trade-off for the neutral category, with the base model showing a very bad recall but better precision, while codeml shows a bad precision with a good recall.

This drop in performance for the base model is also seen in the calibration of the probabilities (Fig. 9). In particular, the base model is now overly conservative in its prediction of sites in the positive selection category.

Discussion

In this manuscript, we have presented our work to develop and train a neural network to predict selective regimes at amino acid alignment sites. We have characterized how the performance of the network varied with features of the network architecture and with characteristics of the training data. These data can be used as guidance for training these types of neural networks for site-wise inference on multiple sequence alignments. In particular, we found that varying the embedding size, the number of attention blocks or attention heads had little effect on performance. However, handling pairs of sequences and using larger training datasets improved our results. These two approaches however have shortcomings: training on a larger dataset entails a larger computational footprint, and the model working with pairs of sequences both requires longer training and has a much larger memory footprint than the model working with sequences.

The results obtained by the base model on simulated data are very promising. Comparison with codeml showed that the network achieved equal or better

performance, at a fraction of the computing time. This is surprising because codeml uses codon sequences, while the base model does not have access to synonymous changes, which provide information on the neutral rate of evolution. This suggests that a similar network, trained on codon sequences, could achieve even higher performance. One particularity of our simulations is that they rely on the mutation-selection model, not on a d_N/d_S model. This could cause a decreased performance of codeml, as the simulated data does not fit its underlying model. However, the mutation-selection model appears to be more realistic than d_N/d_S models [4, 13]. Our results could therefore be indicative of expected performances on empirical data. In addition, the probabilities output by the network were better calibrated than the Bayes Empirical Bayes estimates provided by codeml [20]. However, the results depended on whether the test data matched the training data. While the base model obtained much better results than codeml on test data that matched its training, performance was reduced when the test data was simulated with different amino acid profiles.

Our results obtained with different amino acid profiles, or with different phylogenies, show that the performance of the model are very dependent on the match between training and testing data. The training of a neural network on simulations amounts to building a model that performs Bayesian inference [10]. The resulting network therefore encapsulates a prior distribution over all parameters of the simulation model, which can result in altered performance if there are mismatches between the model that generated the training data and the model that generated the test data. This means that when analyzing empirical data, much attention must be devoted to generating realistic data for the training. Here we have anchored the tree diameters on an empirical tree distribution, and the amino acid profiles on empirical data. Unfortunately however, it is certain that the empirical distributions that we have used do not cover the whole range of parameter values that would be needed to fit different types of alignments. Alignments from different groups of species will differ in their size, their evolutionary depth, their GC content, or the selective pressures that have applied on them. Training a network that could perform equally well on any type of biological alignment is an important challenge. Perhaps a feasible approach would be to train several networks to cover different areas of the parameter space, and develop a method to choose the most relevant network depending on characteristics of the alignment under study. Another solution would be to train a network *de novo* for a given dataset. Our results show that the training of our network on a GPU plus the inference on 10 alignments on a single CPU took about the same amount of time as running codeml on these 10 alignments on 10 CPUs. When inference needs to be performed at the genomic scale, for thousands of gene alignments, training de novo, on simulations that resemble the empirical data, provides a lower footprint than using codeml. Our open source code can be deployed for such a use-case.

This work further suggests that similar neural networks could be used to predict other features of amino acid sites. For instance, a network could be used to predict sites undergoing shifts in the direction of selection associated to phenotype changes [2]. To do so, the network should take as additional input

phenotypic states at each leaf of the tree, or a phylogenetic tree annotated with phenotypic states at each node. Beyond site-wise prediction, similar networks could also predict model parameters that apply to an entire alignment, such as a matrix of nucleotide substitutions, or an average d_N/d_S. Because the transformer architecture at the heart of our model can capture dependencies between sites of the alignment, one could also design a model to learn parameters of models that capture co-evolution between sites, such as the Direct Coupling Analysis model [8].

Conclusion

In this work, we have shown that a neural network trained on simulated amino acid sequences could predict selective pressures acting on individual protein sites. It performed better than a state-of-the-art method across datasets that resemble its training data, but can perform less well when there is a difference between training and test data. Looking ahead, this shows that a similar network working on codon sequences, and trained on data that resembles empirical data, could markedly outperform maximum likelihood approaches to characterize selective pressures acting on sites. Further, it paves the way for the inference of other aspects of molecular evolution.

Acknowledgements

This work was funded by the Agence Nationale de la Recherche (Deelogeny, ANR-23-CE45-0027). This work was performed using the computing facilities of the CC LBBE/PRABI and of the CC IN2P3.

References

[1] Nadav Brandes et al. "Genome-wide prediction of disease variant effects with a deep protein language model". en. In: *Nature Genetics* 55.9 (Sept. 2023). Publisher: Nature Publishing Group, pp. 1512–1522. ISSN: 1546-1718. DOI: 10.1038/s41588-023-01465-0. URL: http://www.nature.com/articles/s41588-023-01465-0 (visited on 06/28/2024).

[2] Louis Duchemin et al. "Evaluation of Methods to Detect Shifts in Directional Selection at the Genome Scale". In: *Molecular Biology and Evolution* 40.2 (Feb. 2023), msac247. ISSN: 1537-1719. DOI: 10.1093/molbev/msac247. URL: https://doi.org/10.1093/molbev/msac247 (visited on 06/20/2023).

[3] Dimitri Höhler et al. "RAxML Grove: an empirical phylogenetic tree database". In: *Bioinformatics* 38.6 (Dec. 2021), pp. 1741–1742. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btab863. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8896636/ (visited on 05/02/2022).

[4] Christopher T. Jones, Edward Susko, and Joseph P. Bielawski. "Looking for Darwin in Genomic Sequences: Validity and Success Depends on the Relationship Between Model and Data". en. In: *Evolutionary Genomics: Statistical and Computational Methods*. Ed. by Maria Anisimova. Methods in Molecular Biology. New York, NY: Springer, 2019, pp. 399–426. ISBN: 978-1-4939-9074-0. DOI: 10.1007/978-1-4939-9074-0_13. URL: https://doi.org/10.1007/978-1-4939-9074-0_13 (visited on 10/13/2021).

[5] John Jumper et al. "Highly accurate protein structure prediction with AlphaFold". eng. In: *Nature* 596.7873 (Aug. 2021), pp. 583–589. ISSN: 1476-4687. DOI: 10.1038/s41586-021-03819-2.

[6] Thibault Latrille et al. "Estimating the proportion of beneficial mutations that are not adaptive in mammals". en. In: *PLOS Genetics* 20.12 (Dec. 2024). Publisher: Public Library of Science, e1011536. ISSN: 1553-7404. DOI: 10.1371/journal.pgen.1011536. URL: https://journals.plos.org/plosgenetics/article?id=10.1371/journal.pgen.1011536 (visited on 03/25/2025).

[7] Yu K. Mo, Matthew Hahn, and Megan L. Smith. "Applications of Machine Learning in Phylogenetics". en. In: (Oct. 2023). Publisher: EcoEvoRxiv. URL: https://ecoevorxiv.org/repository/view/6096/ (visited on 10/20/2023).

[8] Faruck Morcos et al. "Direct-coupling analysis of residue coevolution captures native contacts across many protein families". In: *Proceedings of the National Academy of Sciences* 108.49 (Dec. 2011). Publisher: Proceedings of the National Academy of Sciences, E1293–E1301. DOI: 10.1073/pnas.1111471108. URL: http://www.pnas.org/doi/10.1073/pnas.1111471108 (visited on 03/25/2025).

[9] Luca Nesterenko, Bastien Boussau, and Laurent Jacob. "Phyloformer: towards fast and accurate phylogeny estimation with self-attention networks". en. In: *bioRxiv* (June 2022). DOI: 10.1101/2022.06.24.496975. URL: http://biorxiv.org/lookup/doi/10.1101/2022.06.24.496975 (visited on 10/13/2022).

[10] Luca Nesterenko et al. "Phyloformer: Fast, accurate and versatile phylogenetic reconstruction with deep neural networks". en. In: *bioRxiv* (2024). DOI: https://doi.org/10.1101/2024.06.17.599404. URL: https://www.biorxiv.org/content/10.1101/2024.06.17.599404v1.

[11] Luca Nesterenko et al. "Phyloformer: Fast, Accurate, and Versatile Phylogenetic Reconstruction with Deep Neural Networks". In: *Molecular Biology and Evolution* 42.4 (Mar. 2025), msaf051. ISSN: 0737-4038. DOI: 10.1093/molbev/msaf051. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11965795/ (visited on 05/09/2025).

[12] Simon Penel et al. "Databases of homologous gene families for comparative genomics". en. In: *BMC Bioinformatics* 10.Suppl 6 (Jan. 2009), S3. DOI: 10.1186/1471-2105-10-S6-S3. URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2697650/.

[13] Nicolas Rodrigue, Thibault Latrille, and Nicolas Lartillot. "A Bayesian Mutation–Selection Framework for Detecting Site-Specific Adaptive Evolution in Protein-Coding Genes". In: *Molecular Biology and Evolution* 38.3 (Oct. 2020), pp. 1199–1208. ISSN: 0737-4038. DOI: 10.1093/molbev/msaa265. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7947879/ (visited on 10/13/2021).

[14] Nicolas Rodrigue, Hervé Philippe, and Nicolas Lartillot. "Mutation-selection models of coding sequence evolution with site-heterogeneous amino acid fitness profiles". en. In: *Proceedings of the National Academy of Sciences* 107.10 (Mar. 2010), pp. 4629–4634. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0910915107. URL: http://www.pnas.org/content/107/10/4629 (visited on 09/27/2018).

[15] David Silver et al. "Mastering the game of Go without human knowledge". en. In: *Nature* 550.7676 (Oct. 2017), pp. 354–359. ISSN: 0028-0836. DOI: 10.1038/nature24270. URL: https://www.nature.com/nature/journal/v550/n7676/full/nature24270.html.

[16] Daniele Silvestro, Thibault Latrille, and Nicolas Salamin. "Improved estimation of molecular evolution coupling stochastic simulations and deep learning". en. In: *bioRxiv* (2023). DOI: https://doi.org/10.1101/2023.09.08.556821. URL: https://www.biorxiv.org/content/10.1101/2023.09.08.556821v1.

[17] Asif U. Tamuri and Mario dos Reis. "A mutation-selection model of protein evolution under persistent positive selection". en. In: *bioRxiv* (May 2021). Publisher: Cold Spring Harbor Laboratory Section: New Results, p. 2021.05.18.444611. DOI: 10.1101/2021.05.18.444611. URL: https:

//www.biorxiv.org/content/10.1101/2021.05.18.444611v1 (visited on 06/01/2021).

[18] Jeffrey L. Thorne, Hirohisa Kishino, and Joseph Felsenstein. "An evolutionary model for maximum likelihood alignment of DNA sequences". en. In: *Journal of Molecular Evolution* 33.2 (Aug. 1991), pp. 114–124. ISSN: 1432-1432. DOI: 10.1007/BF02193625. URL: https://doi.org/10.1007/BF02193625 (visited on 07/01/2019).

[19] Ziheng Yang. "PAML 4: phylogenetic analysis by maximum likelihood". eng. In: *Molecular Biology and Evolution* 24.8 (Aug. 2007), pp. 1586–1591. ISSN: 0737-4038. DOI: 10.1093/molbev/msm088.

[20] Ziheng Yang, Wendy S.W. Wong, and Rasmus Nielsen. "Bayes Empirical Bayes Inference of Amino Acid Sites Under Positive Selection". In: *Molecular Biology and Evolution* 22.4 (Apr. 2005), pp. 1107–1118. ISSN: 0737-4038. DOI: 10.1093/molbev/msi097. URL: https://doi.org/10.1093/molbev/msi097 (visited on 03/25/2025).