Predicting interaction partners using masked language modeling

Umberto Lupo*
School of Life Sciences, Institute of Bioengineering
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015, Lausanne, Switzerland
umberto.lupo@epfl.ch

Damiano Sgarbossa*
School of Life Sciences, Institute of Bioengineering
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015, Lausanne, Switzerland
damiano.sgarbossa@epfl.ch

Anne-Florence Bitbol
School of Life Sciences, Institute of Bioengineering
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015, Lausanne, Switzerland
anne-florence.bitbol@epfl.ch

Abstract

Determining which proteins interact together from their amino acid sequences is an important task. In particular, even if an interaction is known to exist in some species between members of two protein families, determining which other members of these families are interaction partners can be tricky. Indeed, it requires identifying which paralogs interact together. Various methods have been proposed to this end. Here, we present a new one, which relies on a protein language model trained on multiple sequence alignments and directly exploits the fact that this model was trained to fill in masked amino acids. We obtain promising results on two different benchmark pairs of interacting protein families where partners are known. In particular, performance is good even for shallow alignments, while previous coevolution-based methods require deep ones. Performance is also found to quickly improve by giving the model correct examples of interacting sequences.

1 Introduction

Mapping functional protein-protein interactions is an important question in cell and systems biology. High-throughput experiments capable of resolving protein-protein interactions remain challenging [1], even for model organisms. Meanwhile, statistical and machine learning methods trained on an ever increasing amount of data have been developed to find contacts between known interaction partners from sequences [2–9], and predict functional interaction partners from sequence [10–13] and/or structure data [14–16] and/or molecular surface descriptors on structures [17].

*These authors contributed equally to this work

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Paralog matching is the problem of matching interaction partners correctly among the paralogous proteins belonging to two interacting families. Aside from further elucidating the nature and evolutionary history of protein-protein interaction networks, addressing this problem for a given pair (or set) of interacting protein families allows for the construction of a concatenated (joint) multiple sequence alignment (MSA) which can then be used, in addition to a specific sequence query, as input to a structure prediction model for protein complexes such as AlphaFold-Multimer [15, 18] (but see also [19]). The quality of the pairings in this MSA strongly affects the accuracy of the predicted structure for the query complex [19], making the paralog matching problem an integral part of the structure prediction workflow for heteropolymers. In the case of prokaryotes, genes of interacting proteins are frequently colocalized in operons and can therefore often be matched by chromosomal vicinity with good confidence, a method which is often used in practice [2, 5]. However, even in prokaryotes, many important interactions exist across distinct operons [20, 21]. Furthermore, interaction partners in eukaryotes are generally *not* encoded in close genomic locations. Thus, paralog matching remains an open problem in general.

Aside from genome colocalisation, methods for addressing this problem include phylogenetic profiling [22–24], exploiting similarities between phylogenetic trees of groups of orthologous proteins [25–30] and relying on orthology, determined by reciprocal closest matching sequences [13–16, 31]. Other coevolutionary methods make use of full amino acid sequences and rely upon the existence of correlations between residues of interacting proteins [10, 11, 32], due both to the need to maintain physico-chemical complementarity among amino acids in contact, and to shared evolutionary history, which is in fact quite useful for the pairing [33–35]. However, the performance of these models is fundamentally limited on small alignments, as they require accurate estimation of two-body statistics for all pairs of residue positions.

The core idea behind coevolution-based methodologies is to single out, among all possible sets of complete pairings between interacting paralogs, those which maximise a global coevolutionary signal in the candidate joint MSA. Building on this principle, while attempting to overcome the aforementioned limitations with small alignments, we propose using recently developed protein language models that take MSAs as inputs [36, 37]. These models are able to directly exploit the covariation signal, while also benefiting from training over large sequence databases containing a large number of diverse protein families. Thus motivated, we focus on MSA Transformer [36], a protein language model which was trained on MSAs using the masked language modeling (MLM) objective, and present a differentiable pipeline for optimizing paralog matchings using the MLM loss.

2 Methods

We use the pre-trained 100-million-parameter MSA Transformer model [36], which takes MSAs as inputs and was trained with a variant of the masked language modeling (MLM) objective [38], on a training set of 26 million monomer MSAs constructed from UniRef50 clusters. The model's training objective was to correctly predict the identity of randomly masked residue positions in the MSAs in its training set. Its MLM loss for an MSA $\mathcal{M}$, and its masked version $\widetilde{\mathcal{M}}$, is:

$$\mathcal{L}_{\text{MLM}}(\mathcal{M}, \widetilde{\mathcal{M}}; \theta) = -\sum_{(m,i) \,\in\, \text{mask}} \log p(x_{m,i} \,|\, \widetilde{\mathcal{M}}; \theta)\,. \tag{1}$$

Here, $x_{m,i}$ denotes the amino acid at the i-th residue position in the m-th sequence of $\mathcal{M}$, while θ stands for all the model parameters. At each residue position in the input MSA, MSA Transformer outputs a vector of probabilities for each of the 21 possible amino-acid and gap symbols, and $p(x_{m,i} \,|\, \widetilde{\mathcal{M}}; \theta)$ in Eq. (1) is the probability associated with the correct residue $x_{m,i}$ at MSA position (m, i). MSA Transformer's interleaving of multi-headed (tied) row attention blocks and (untied) column attention blocks, over several layers, implies that the accessible context for a masked token consists not only of amino acids at different positions along the same sequence, but also of amino acids from other sequences [36, 39].

After pre-training, each summand in the right-hand side of Eq. (1) can be interpreted as the model's estimate of the (negative) log-likelihood of the amino acid $x_{m,i}$ at a masked position (m, i) [40–42]. We phrase paralog matching for a pair of protein families as the problem of concatenating a pair of MSAs, each one corresponding to one of the protein families and containing several paralogs per species, so that correct interaction partners are placed on the same row of the concatenated MSA.

In this context, we posit that the trained MSA Transformer model should regard randomly masked residues in correctly concatenated MSAs as more likely, on average, than randomly masked residues in incorrectly concatenated MSAs. For proteins families that actually interact, this holds true despite the fact that MSA Transformer was only trained on monomeric MSAs, because coevolutionary signal is present in these MSAs and is accessible to the model, as we show in Fig. A1. Hence, we set out to find interaction partners by looking for pairings that minimise a suitably constructed MLM loss. Indeed, we show in Fig. A2 that the MLM loss decreases as the fraction of correctly matched sequences increases.

For a pair of interacting MSAs, a brute-force search through all possible in-species one-to-one matchings would scale factorially in the size of each species. Instead, we formulate a differentiable optimization problem that can be more efficiently solved, using gradient methods, to yield configurations minimizing our MLM loss. Note that one-to-one matchings can be encoded as permutation matrices. We exploit the fact, shown in [43], that permutation matrices can be approximated arbitrarily well by using the so-called *Sinkhorn operator* S, which is defined on square matrices X as follows:

$$S(X) = \lim_{l\to\infty} S^l(X), \quad \text{where} \quad S^l(X) = (\mathcal{T}_\mathrm{c} \circ \mathcal{T}_\mathrm{r})^l(\exp(X)), \tag{2}$$

$\mathcal{T}_\mathrm{c}$ and $\mathcal{T}_\mathrm{r}$ are the row- and column-wise normalization operators, and $\exp$ denotes the component-wise matrix exponential.[2] The set $\mathcal{P}_N$ of permutation matrices of N objects can be parametrized exactly by square matrices X via the *matching operator*

$$M(X) = \arg\max_{P\in\mathcal{P}_N} [\mathrm{trace}(P^\mathrm{T} X)], \tag{3}$$

which can be computed using standard non-differentiable algorithms for linear assignment problems [44].[3] The aforementioned approximation result is then that $M(X) = \lim_{\tau\to 0^+} S(X/\tau)$ for almost all X [43, Theorem 1]. Hence, by choosing a suitably small value of τ, and using S^l [Eq. (2)] instead of S for a suitably large l, we can define a smooth mapping $\hat{S}$ which sends arbitrary square matrices to "soft permutations" approximating *bona fide* ("hard") permutations. In our experiments, we use $\tau = 1$ and $l = 10$. Applying general soft permutations directly on an MSA (after one-hot encoding its residues) yields a dataset of consisting of "amino acid mixtures" at each MSA position. Such datasets are out of distribution relative to MSA Transformer's pre-training. Besides, we wish to optimize for an MLM loss defined on realistic MSAs. In order to be able to backpropagate through $\hat{S}$, while also evaluating MLM losses only on MSAs shuffled by hard permutations, we compute the full matching operator M [Eq. (3)] in the forward pass, but propagate gradients backwards through $\hat{S}$ alone.[4]

Our use of a language model allows for contextual conditioning, a common technique in natural language processing. Indeed, if any correctly paired sequences are already known, they can be included as part of the joint MSA input to MSA Transformer. In this case, we exclude their pairing from the optimization process – in particular, by not masking any of their amino acids. We call these known paired sequences "positive examples" and, as we show in Results, the presence of even just a few of them can lead to large gains in accuracy.

Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be MSAs of interacting protein families, consisting of N_pos positive examples and K unmatched species, each of size N_k where $k = 1, \ldots, K$. Using the tools just described, we optimize a set $\{X_k\}_{k=1,\ldots,K}$ of square matrices, each of size $N_k \times N_k$. Our MLM-based loss for this optimization is defined as follows: (1) perform a shuffle of $\mathcal{M}_1$ relative to $\mathcal{M}_2$ using the permutation matrices corresponding to the current $\{X_k\}_k$ (plus an optional noise term, see below), to obtain a concatenated MSA $\mathcal{M}$; (2) create m distinct masks for $\mathcal{M}$ (excluding any positive example tokens from the masking); (3) compute m losses, given by Eq. (1) for each of the masks, and average them. Importantly, we mask the amino acids of only one of the two MSAs, chosen uniformly at random within it with a high masking probability p (70%).[5] Our rationale for using large masking probabilities is that, in this case, the model is forced to predict masked residues in one of the two MSAs by using information coming mostly from the other MSA – see Fig. A2. The averaging from several different masks helps us achieve smoother loss curves; in practice, to limit the computational burden, we use $m = 4$ throughout.

[2] That is, S^l consists of applying $\exp$ and then iteratively normalizing rows and columns l times.

[3] More precisely, the right-hand side of Eq. (3) has a unique solution for almost all X [43].

[4] See [45] for a similar use of "gradient bypassing" in the context of protein design. We write the hard permutation as $[M(X) - \hat{S}(X)] + \hat{S}(X)$, and halt gradient backpropagation through the term in square brackets.

[5] Uniformly random masking with $p = 15\%$ was used during MSA Transformer's pre-training [36].

Furthermore, following [43], after updating (or initializing) each X_k, we add to it a noise term given by a matrix of standard i.i.d. Gumbel noise multiplied by a scale factor. The addition of noise ensures that the X_k do not get stuck at degenerate values for the right-hand side of Eq. (3), and more generally encourages the algorithm to explore larger regions in the space of permutations. As scale factor for this noise we choose 0.1 times the sample standard deviation of the entries of X_k. Finally, since the matching operator is scale-invariant, we can regularize the matrices X_k to have small Frobenius norm. We find this to be beneficial and implement it through weight decay. Since species in our MSAs do not have a fixed size, for the k-th species in our dataset we optimize X_k with weight decay $w_k = 0.1(N_k/10)^2$. We use the Adam optimizer and a learning rate scheduler consisting of a warm-up period (first half of a "1cycle" policy [46]) bringing the learning rate up to $\lambda_{\max} = 0.1$, followed by a "reduce on loss plateau" learning rate scheduler. In all our runs, we initialize each X_k with zero-mean i.i.d. Gaussian entries ($\sigma = 0.1$), and perform $n = 200$ gradient steps.

We observe that, even though the loss generally converges to a minimum average value during our optimization runs, there are often several distinct hard permutations associated to the smallest loss values. This may indicate a flattening of the loss landscape relative to the inherent fluctuations in the MLM loss, and/or the existence of multiple local minima. To extract a single configuration of matchings from each of our runs, we average the hard permutation matrices associated to the q lowest losses, and evaluate the matching operator [Eq. (3)] on the resulting averages.[6] This yields a single hard permutation for the run. Furthermore, we propose using each entry in these averages as an indicator of the model's confidence in the matching of the corresponding pair of sequences. Indeed, pairs that are present in most low-loss configurations are presumably essential for the optimization process. Accordingly, we refer to such averaged matrices as "confidence matrices".

We also test two methods for improving performance further. In the first method, which we call Multi-Run Aggregation (MRA), we perform N_{runs} optimization runs for each interacting MSA, using independent initializations. Then, we average the hard permutations independently obtained from each run (using lowest losses as explained above), to obtain more reliable confidence matrices and hard permutations. The second method is an iterative procedure analogous to the Iterative Pairing Algorithm (IPA) of Refs. [10, 32]. At the beginning of each run, we determine the pairing with highest confidence according the previous N_{prev} runs, in the same way as for the MRA method (we use $N_{\text{prev}} = 4$). This pairing is promoted to a positive example in all subsequent runs. N_{IPA} runs are performed; in the last one, all remaining unmatched sequences are paired at once.

3 Results

We developed and tested our method using joint MSAs extracted from two datasets. The first dataset is composed of 23,632 cognate pairs of histidine kinases (HK) and response regulators (RR) from the P2CS database [47, 48], paired using genome proximity, and previously described in [10, 32]. HK and RR are interacting protein families from prokaryotic two-component signaling systems. These proteins have a strong specificity with their cognate partners, a large number of homologs, and interaction partners known from genome proximity, which makes them an attractive benchmark dataset. The average size of the species in this dataset is 10.23 (std: 7.85). The second dataset consists of 17,950 ABC transporter protein pairs, homologous to the *Escherichia coli* MALG-MALK pair of maltose and maltodextrin transporters, also paired using genome proximity [5, 10]. The average size of the species in this dataset is 5.68 (std: 5.60).

We fine-tuned all the hyperparameters involved in our algorithm (see Methods) using two joint MSAs of depth ~ 50, constructed by selecting random species from the HK-RR dataset.[7] We then tested our method on new sequences from both the HK-RR dataset and the MALG-MALK dataset. We restricted our experiments to MSAs of depth $\lesssim 50$ because, in this small data regime, we can most effectively leverage MSA Transformer's extensive pre-training, while alternative coevolutionary methods [10, 11, 32] require considerably deeper alignments to achieve good performance. Furthermore, MSA Transformer's large memory footprint provides a constraint on the depth and length of the concatenated alignment.

We tested our MRA and IPA methods for different values of N_{runs} and N_{IPA} (see Methods), on 40 MSAs from the HK-RR dataset and 40 MSAs from the MALG-MALK dataset, constructed in the

[6] We choose $q = 20$ as we typically observe fast convergence.

[7] Species are added one by one until an MSA depth between 45 and 55 is reached.

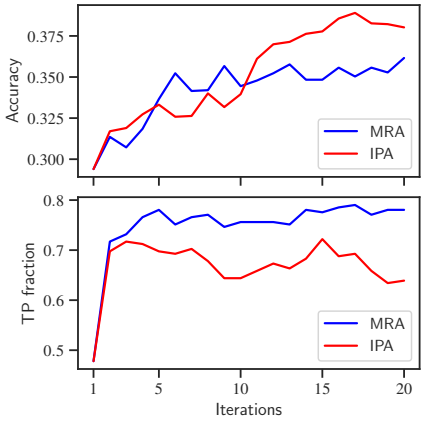
same way as the two HK-RR MSAs used for hyperparameter fine-tuning. We tested different numbers N_{pos} of positive examples consisting of groups of entire species (including $N_{\text{pos}} = 0$, meaning no positive examples). We also tested a special case of the MRA method in which, using 49 positive examples, we optimize matchings for one randomly selected species from the HK-RR dataset at a time. We refer to this as "species-by-species" (SBS) optimization, and repeat it for 300 species, using only one optimization run in each case ($N_{\text{runs}} = 1$).

For each of our experiments, we report two measures of performance, accuracy and TP fraction, as well as for other methods for comparison. Accuracy is defined as the fraction of correct pairs over all predicted ones (here a full one-to-one mapping is predicted), while the TP fraction is the accuracy over the top 10% predicted pairs, when ranked by predicted confidence (see Methods).

Results are shown in Fig. 1(a). They show that our methods perform always better than the null expectation,[8] and that they in fact outperform other coevolution-based methods (DCA-IPA [10], MI-IPA [32] and GA-IPA [35]). They also show that both accuracy and TP fraction increase significantly as more positive examples are used. As can be seen in Fig. 1(b), for the HK-RR dataset and without positive examples, the accuracy for both the MRA and IPA method increases noticeably as N_{runs} and N_{IPA} (respectively) are increased.

MSAs	Type	Pos. Ex.	Iter.	Acc.	TP fr.
HK-RR	Null Model	-	-	0.09	-
MALG-MALK	Null Model	-	-	0.20	-
HK-RR	DCA-IPA [10]	0	-	< 0.2	-
HK-RR	MI-IPA [32]	0	-	< 0.2	-
HK-RR	GA-IPA [35]	0	-	< 0.3	-
HK-RR	MRA	0	20	0.36	0.78
HK-RR	MRA	11	5	0.47	0.96
HK-RR	MRA	19	5	0.59	1.00
HK-RR	MRA	45	5	0.71	1.00
HK-RR	IPA	0	20	0.38	0.64
HK-RR	IPA	11	5	0.46	0.86
HK-RR	IPA	19	5	0.57	0.93
HK-RR	IPA	45	5	0.70	0.98
HK-RR	SBS	49	1	0.68	0.86
MALG-MALK	MRA	0	5	0.45	0.87
MALG-MALK	IPA	0	5	0.42	0.72

(a)



(b)

Figure 1: **Performance of pairing.** (a) We report two measures of performance, accuracy (Acc.) and TP fraction (TP Fr.), as defined in the text, for variants of our methods [MRA, IPA, SBS, with various numbers of positive examples (Pos Ex.) and iterations (Iter.)], as well as for other methods for comparison. (b) For the MRA (blue) and the IPA (red) method and the set of 40 HK-RR MSAs we used for testing, and without using positive examples, we show the dependence of accuracy (top) and TP fraction (bottom) on the number of iterations performed (N_{runs} for MRA, N_{IPA} for IPA).

4 Discussion

Our results demonstrate that our methods yield good performance in the paralog matching problem, on two distinct datasets of interacting protein families. Hence, losses based on masked language modeling can be used to good effect for pairing sequences belonging to interacting families. This differentiates our methods from other recent work [49], in which MSA Transformer is also used to address the paralog matching problem, using column attention matrices [39].

There are several ways in which our methods could be improved. First, for simplicity, we masked only one of the two interacting MSAs in our experiments. Further work is required to assess how to best mask concatenated MSAs. Second, our methods could be combined with complementary approaches, allowing e.g. a better initialization [35]. Finally, it would be interesting to further test and improve the generalizability of our methods to various other interacting families. In particular, an important application would be on eukaryotic families, which often have many paralogs and cannot be paired by genome proximity.

[8] This is the expected fraction of correctly paired sequences in a random within-species matching.

References

[1] S. V. Rajagopala et al. "The binary protein-protein interaction landscape of Escherichia coli". In: *Nat. Biotechnol.* 32.3 (2014), pp. 285–290.

[2] M. Weigt et al. "Identification of direct residue contacts in protein-protein interaction by message passing". In: *Proc. Natl. Acad. Sci. U.S.A.* 106.1 (2009), pp. 67–72.

[3] A. Procaccini et al. "Dissecting the specificity of protein-protein interaction in bacterial two-component signaling: orphans and crosstalks". In: *PLoS ONE* 6.5 (2011), e19729.

[4] C. Baldassi et al. "Fast and accurate multivariate Gaussian modeling of protein families: predicting residue contacts and protein-interaction partners". In: *PLoS ONE* 9.3 (2014), e92721.

[5] S. Ovchinnikov, H. Kamisetty, and D. Baker. "Robust and accurate prediction of residue-residue interactions across protein interfaces using evolutionary information". In: *eLife* 3 (2014), e02030. DOI: 10.7554/eLife.02030.

[6] T. A. Hopf et al. "Sequence co-evolution gives 3D contacts and structures of protein complexes". In: *eLife* 3 (2014), e03430. DOI: 10.7554/eLife.03430.

[7] S. Tamir et al. "Integrated strategy reveals the protein interface between cancer targets Bcl-2 and NAF-1". In: *Proc. Natl. Acad. Sci. U.S.A.* 111.14 (2014), pp. 5177–5182.

[8] R. N. dos Santos et al. "Dimeric interactions and complex formation using direct coevolutionary couplings". In: *Sci Rep* 5 (2015), p. 13652.

[9] C. Feinauer et al. "Inter-Protein Sequence Co-Evolution Predicts Known Physical Interactions in Bacterial Ribosomes and the Trp Operon". In: *PLoS ONE* 11.2 (2016), e0149166.

[10] A.-F. Bitbol et al. "Inferring interaction partners from protein sequences". In: *Proc. Natl. Acad. Sci. U.S.A.* 113.43 (2016), pp. 12180–12185.

[11] T. Gueudre et al. "Simultaneous identification of specifically interacting paralogs and inter-protein contacts by direct coupling analysis". In: *Proc. Natl. Acad. Sci. U.S.A.* 113.43 (2016), pp. 12186–12191.

[12] Q. Cong et al. "Protein interaction networks revealed by proteome coevolution". In: *Science* 365.6449 (2019), pp. 185–189.

[13] A. G. Green et al. "Large-scale discovery of protein interactions at residue resolution using co-evolution calculated from genomic sequences". In: *Nature Commun.* 12.1 (2021), p. 1396.

[14] I. Humphreys et al. "Computed structures of core eukaryotic protein complexes". In: *Science* 374.6573 (2021). ISSN: 10959203. DOI: 10.1126/science.abm4805.

[15] R. Evans et al. "Protein complex prediction with AlphaFold-Multimer". In: *bioRxiv* (2021). DOI: 10.1101/2021.10.04.463034.

[16] P. Bryant, G. Pozzati, and A. Elofsson. "Improved prediction of protein-protein interactions using AlphaFold2". In: *Nat. Commun.* 13.1 (2022), p. 1265.

[17] P. Gainza et al. "Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning". In: *Nat. Methods* 17.2 (2020), pp. 184–192. ISSN: 15487105. DOI: 10.1038/s41592-019-0666-6.

[18] M. Mirdita et al. "ColabFold: making protein folding accessible to all". In: *Nat. Methods* 19.6 (2022), pp. 679–682. DOI: 10.1038/s41592-022-01488-1.

[19] P. Bryant, G. Pozzati, and A. Elofsson. "Improved prediction of protein-protein interactions using AlphaFold2". In: *Nature Communications* 13.1 (2022), p. 1265. DOI: 10.1038/s41467-022-28865-w.

[20] S. Orchard et al. "The MIntAct project—IntAct as a common curation platform for 11 molecular interaction databases". In: *Nucleic Acids Research* 42.D1 (2013), pp. D358–D363. ISSN: 0305-1048. DOI: 10.1093/nar/gkt1115. eprint: https://academic.oup.com/nar/article-pdf/42/D1/D358/3585170/gkt1115.pdf.

[21] J. M. Peters et al. "A Comprehensive, CRISPR-based Functional Analysis of Essential Genes in Bacteria". In: *Cell* 165.6 (2016), pp. 1493–1506. ISSN: 0092-8674. DOI: https://doi.org/10.1016/j.cell.2016.05.003.

[22] M. Pellegrini et al. "Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles". In: *Proceedings of the National Academy of Sciences* 96.8 (1999), pp. 4285–4288. DOI: 10.1073/pnas.96.8.4285.

[23] G. Croce et al. "A multi-scale coevolutionary approach to predict interactions between protein domains". In: *PLoS Comput. Biol.* 15.10 (2019), e1006891.

[24] D. Moi et al. “Scalable phylogenetic profiling using MinHash uncovers likely eukaryotic sexual reproduction genes”. In: *PLoS Comput. Biol.* 16.7 (2020), e1007553.

[25] F. Pazos and A. Valencia. “Similarity of phylogenetic trees as indicator of protein–protein interaction”. In: *Protein Engineering, Design and Selection* 14.9 (2001), pp. 609–614. ISSN: 1741-0126. DOI: 10.1093/protein/14.9.609. eprint: https://academic.oup.com/peds/article-pdf/14/9/609/18546034/140609.pdf.

[26] D. Juan, F. Pazos, and A. Valencia. “High-confidence prediction of global interactomes based on genome-wide coevolutionary networks”. In: *Proceedings of the National Academy of Sciences* 105.3 (2008), pp. 934–939. DOI: 10.1073/pnas.0709671105.

[27] R. Jothi, M. G. Kann, and T. M. Przytycka. “Predicting protein-protein interaction by searching evolutionary tree automorphism space”. In: *Bioinformatics* 21 Suppl 1 (2005), pp. i241–250.

[28] S. Bradde et al. “Aligning graphs and finding substructures by a cavity approach”. In: *EPL* 89.3 (2010).

[29] D. Ochoa and F. Pazos. “Studying the co-evolution of protein families with the Mirrortree web server”. In: *Bioinformatics* 26.10 (2010), 1370–1371, http://csbg.cnb.csic.es/mtserver.

[30] D. Ochoa et al. “Detection of significant protein coevolution”. In: *Bioinformatics* 31.13 (2015), 2166–2173, http://csbg.cnb.csic.es/pMT/.

[31] Q. Cong et al. “Protein interaction networks revealed by proteome coevolution”. In: *Science* 365.6449 (2019), pp. 185–189.

[32] A.-F. Bitbol. “Inferring interaction partners from protein sequences using mutual information”. In: *PLoS Comput. Biol.* 14.11 (2018), e1006401.

[33] G. Marmier, M. Weigt, and A.-F. Bitbol. “Phylogenetic correlations can suffice to infer protein partners from sequences”. In: *PLoS Comput. Biol.* 15.10 (2019), e1007179.

[34] A. Gerardos, N. Dietler, and A.-F. Bitbol. “Correlations from structure and phylogeny combine constructively in the inference of protein partners from sequences”. In: *PLoS Comput Biol* 18.5 (2022), e1010147.

[35] C. A. Gandarilla-Perez et al. “Combining phylogeny and coevolution improves the inference of interaction partners among paralogous proteins”. In: *bioRxiv* (2022). DOI: 10.1101/2022.08.24.505105.

[36] R. M. Rao et al. “MSA Transformer”. In: *Proceedings of the 38th International Conference on Machine Learning*. Proceedings of Machine Learning Research 139 (2021), pp. 8844–8856.

[37] J. Jumper et al. “Highly accurate protein structure prediction with AlphaFold.” In: *Nature* 596 (2021), pp. 583–589.

[38] J. Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[39] U. Lupo, D. Sgarbossa, and A.-F. Bitbol. “Protein language models trained on multiple sequence alignments learn phylogenetic relationships”. In: *Nat. Commun.* 13.6298 (2022). DOI: 10.1038/s41467-022-34032-y.

[40] A. Wang and K. Cho. “BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model”. In: *CoRR* abs/1902.04094 (2019). arXiv: 1902.04094.

[41] K. Goyal, C. Dyer, and T. Berg-Kirkpatrick. “Exposing the Implicit Energy Networks behind Masked Language Models via Metropolis–Hastings”. In: *arXiv* 10.48550/arxiv.2106.02736 (2021).

[42] R. Rao et al. “Transformer protein language models are unsupervised structure learners”. In: *International Conference on Learning Representations*. 2021.

[43] G. E. Mena et al. “Learning latent permutations with Gumbel-Sinkhorn networks”. In: *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings* (2018), pp. 1–22. arXiv: 1802.08665.

[44] H. W. Kuhn. “The Hungarian Method for the assignment problem”. In: *Naval Research Logistics Quarterly* 2 (1955), pp. 83–97.

[45] C. Norn et al. “Protein sequence design by conformational landscape optimization”. In: *Proceedings of the National Academy of Sciences* 118.11 (2021), e2017228118. DOI: 10.1073/pnas.2017228118.

[46] L. N. Smith and N. Topin. *Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates*. 2017. DOI: 10.48550/arXiv.1708.07120.

[47] M. Barakat et al. “P2CS: a two-component system resource for prokaryotic signal transduction research”. In: *BMC Genomics* 10 (2009), p. 315.

[48] M. Barakat, P. Ortet, and D. E. Whitworth. “P2CS: a database of prokaryotic two-component systems”. In: *Nucleic Acids Res.* 39.Database issue (2011), pp. D771–776.

[49] B. Chen et al. “Improve the Protein Complex Prediction with Protein Language Models”. In: *bioRxiv* (2022). DOI: 10.1101/2022.09.15.508065.

A1 Contact maps of concatenated MSAs

To understand whether the protein language model MSA Transformer has learned some notion of interacting partners, we perform a simple experiment: we feed to the model an input MSA made of the concatenation of two paired MSAs of interacting sequences (i.e. interacting sequences are joined in the same row) and we compare the output contact maps with the contacts predicted for an input made of the same MSAs but concatenated with randomly shuffled pairs (so that interacting sequences are typically no longer in the same row of the joint MSA).

In Fig. A1 we observe that MSA Transformer is able to correctly predict the inter-protein contacts when given as input a concatenated MSA made of correctly matched sequences. Instead, if the model is given as input a concatenation of the same MSAs whose rows have been previously shuffled, it is not able to recover the inter-protein contact map (even if it correctly recovers correctly the intra-protein contact maps).

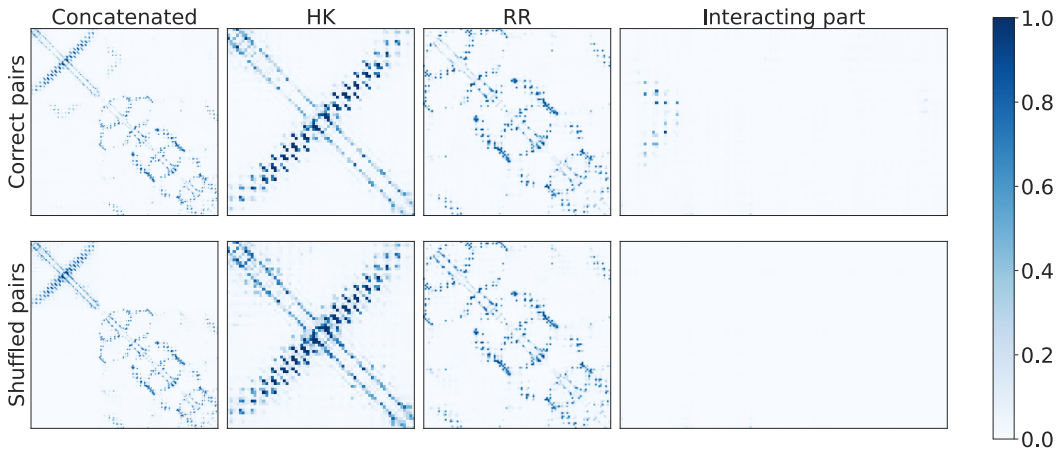


Figure A1: Comparison of contact maps predicted by MSA Transformer for the correct concatenation of an HK MSA and an RR MSA ("Correct pairs"), and for an incorrect concatenation ("Shuffled pairs").

These results suggests that MSA Transformer can distinguish between interacting and non-interacting pairs of protein sequences, despite the fact that dimers (written as sequence concatenations or otherwise) were not in the training set used for its MLM pre-training [36].

A2 Suitability of the Masked Language Modeling loss

To go beyond the qualitative results of Fig. A1 and define a method for identifying interacting protein pairs, we need a way to quantify the correctness of the chosen pairs. In practice, we would like a score that is minimal when all matches are correct and monotonically decreases for increasing number of correct pairs. A natural choice for this score is the loss used in MSA Transformer's pre-training, i.e. the masked language modeling (MLM) loss Eq. (1). As explained in Methods, we adapt the MLM objective to our task by using a slightly modified masking process in which only one of the two concatenated MSAs is masked.

In Fig. A2, we show that this modified loss decreases for increasing numbers of correctly matched sequences in the MSA. We see that the sweet spot of the masking probability p (i.e. the value that gives steeper and smoother loss curves) is at moderately high values ($0.4 \leq p \leq 0.7$). As we also explain in Methods, high masking probabilities make it more challenging for the model to predict the masked amino acids using only information coming from the masked MSA, thus encouraging it to use, instead, information coming from the matched MSA. In this way, we make the value of the loss more sensitive to the correctness of the matching. For these reasons, in all our experiments we use a masking probability of $p = 0.7$.

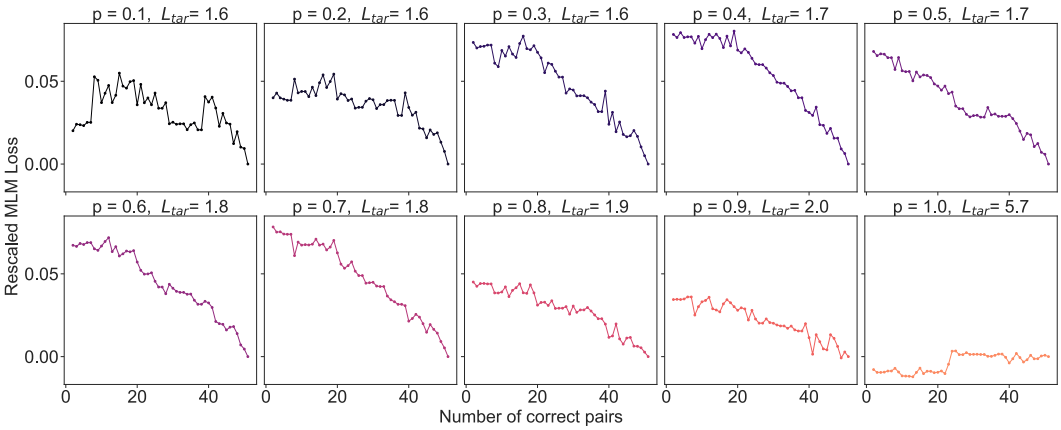


Figure A2: MLM loss vs. number of correct pairs for different masking probabilities. We use an MSA of $M = 50$ sequences and 5 different species. To estimate the expected loss accurately, we used 20 different masks at each step. L_{tar} denotes the expected loss when all pairs are correctly matched. For visualization purposes, in every plot we rescale the loss by shifting it by L_{tar}.