A Appendix

A.1 Methodology Details.

For protein-level prediction tasks, we extract representation features from the [CLS] token of the last layer of ESM-1b and MSA-Transformer and add a new linear layer on top of it. Since there is no [CLS] token appended in AlphaFold, we average all residue embeddings from the last layer and add the same linear layer on top. For residue-level supervised prediction tasks (i.e. SS prediction), we add a linear layer on top of each residue embedding the last layer in all three PLMs. We train these models for each non-zero-shot task by fine-tuning all parameters, including both the new linear layer and the backbone representation model.

For contact prediction, we perform experiments differently for the three models given that the pairwise distance matrix of Evoformer can be directly extracted. For MSA-Transformer and ESM-1b, we follow the [28], extracting and normalizing the attention map from all hidden layers and then training a linear layer on these 2D maps and performing the regression task.

For zero-shot fitness prediction, MSAs are searched in the BFD dataset [23] using Jackhmmer [22] with the default parameters. For annotation prediction tasks, we search MSAs from Uniref90 [42] database with Jackhmmer. For SS and contact prediction, we use the MSAs provided by [35]. MSAs are not used in the two supervised fitness prediction tasks because all sequences are highly similar with only a few positions different. We report all results of Evoformer without template to avoid information leaking for the structure tasks.

A.2 ESM-MSA details

A.2.1 Training set construction

ESM-MSA is essentially a two-tower-based network following sentence-BERT [31] where each tower is represented by the ESM-1b encoder. Unlike sentence-BERT, we have developed an effective negative sampling method to choose more informative negative protein sequences rather than perform random sampling. The schematic of ESM-MSA is shown in Figure 6 and more details are given below:

We first collect high-quality homologous sequence datasets. We use the public trRosetta training set[§] as the ground truth data in this work. Define $H = \{h_1, h_2, ..., h_n\}$ as the entire protein sequence data set, where h_i represents the aligned homologous sequences for protein i, usually called an MSA. Define $h_i = \{q_1, q_2, ..., q_n\}$, where q_i is an individual sequence in a same homologous family. Then define $D_{neg} = \{d_1, d_2, ..., d_n\}$ as the data for negative sampling, where d_i also denotes an individual protein sequence. We use Uniclust30[¶] for sampling non-homologous sequences, which include 200 million individual protein sequences. We then construct a training set including homologous sequence pairs and non-homologous sequence pairs to train the model. Specifically, for a protein sequence q_i belonging to set h_i, we randomly sample its homologous sequence $q_{pos} \in h_i$ from the same set

[§] https://github.com/gjoni/trRosetta
[¶] https://uniclust.mmseqs.com/

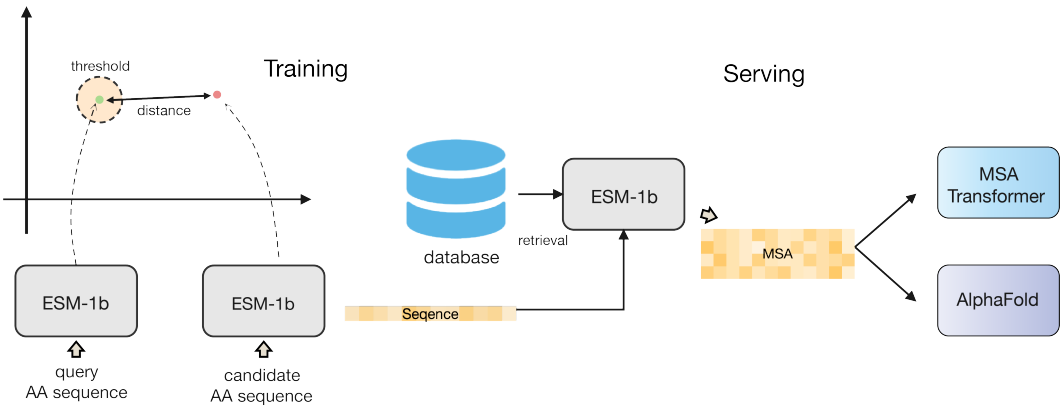

Figure 6: Training and serving schematic of ESM-MSA.

and calculate the biological identity $s \in [0, 1]$ between the two sequences. Finally, we construct a positive sample pair in the training set p^{pos} . Similarly, for the original sequence $q_i \in h_i$, we sample a sequence $q_{neg} \in D$ from the Uniclust30 database as a negative sample(non-homologous sequence), thus constructing a negative sample pair P^{neg} . We finally obtain the set of positive pairs $S_{pos} = \{p_1^{pos}, p_2^{pos}, ..., p_n^{pos}\}$, and the set of negative pairs $S_{neg} = \{p_1^{neg}, p_2^{neg}, ..., p_m^{neg}\}$, and then obtain the complete training set $S = S_{pos} \bigcup S_{neg}$.

A.2.2 Dynamic negative sampling

In this paper, we use dynamic negative sampling at the training stage, a.k.a. hard negative sampling [46]. That is, instead of randomly sampling sequences, we continuously select samples that are more difficult for the model to distinguish. By this way, the model convergence speed is greatly accelerated and the performance of the model is largely improved. Specifically, for the newly initialized model, we randomly select a batch of samples from the database as negative samples for training. After that, in every new round, for each sequence q_i, we randomly sample n sequences from the database and calculate the Euclidean distance between the original sequence and sampled sequences by the model. We select the sampled sequence with the closest Euclidean distance to the original sequence representation as the negative sample. Since the closest Euclidean distance means that the model is not able to distinguish it as negative. To reduce the computational cost, we use a sequence pooling approach, i.e., we first randomly sample N sequences($N \gg n$) from the database as a sub-database, and then perform subsequent sampling operations on such sub-database after the corresponding scores are computed by the model.

A.2.3 Objective Function

For the positive sample pair, the loss function is defined as follows:

$$Loss(p, q)_{pos} = (dist(p, q) - (1 - s))^2$$

where p is the original sequence, q is the homologous sequence of p , and s is the biological identity of the two sequences. The definition is based on the hypothesis that the greater the biological sequence identity of the homologous sequences, the closer the Euclidean distance between the two should be through the representations given by the model. This means that between homologous sequences, the Euclidean distance also varies according to the biological sequence identity, and thus the model can obtain better generalization performance. Therefore, we used the mean square error loss function. Because $s \in [0, 1]$, setting the distance threshold to 1 not only conforms to Occam's razor principle [9], but also fits well with the loss function.

For the negative sample pair, the loss function is defined as follows:

$$Loss(p, q)_{neg} = -logz(p, q)$$

where $z(p, q) = \frac{e^{dist(p,1)}}{e^{dist(p,q)} + e^t}$, t is distance threshold. Actually, this is Cross Entropy function. For non-homologous sequences, we want the model to keep increasing the Euclidean distance between p and q, so it is reasonable to use the distance threshold t as a reference for the Cross Entropy, so that the Euclidean distance of the representations keeps moving away from the distance threshold.

Table 9: Contact map prediction, Precision@L, L/5, L/2.

Model	Precision@L	Precision@$L/2$	Precision@$L/5$
ESM-1b	0.540	0.668	0.783
MSA Transformer	0.660	0.784	0.872
Evoformer	**0.946**	**0.970**	**0.978**

A.2.4 MSA Retrieval and Alignment

First, we use the embedded vector of 'CLS' in ESM-1b (trained by the above approach) as the protein representation. We calculate all protein sequences in the database. Then we use Faiss [21], the library for quick embedding searching, to retrieve homologous sequences. Specifically, we calculate the Euclidean distance between sequence embeddings and keep those under the threshold as homologous to the query protein. One we obtain a set of homologous sequences, we use Famsa[11] to efficiently align them and output an a3m format file.

A.3 HHblits setting

A.3.1 Search the BFD database HHblits setting

n_iter: int = 3, e_value: float = 0.001, maxseq: int = 1000000, realign_max: int = 100000, maxfilt: int = 100000, min_prefilter_hits: int = 1000, all_seqs: bool = False, p: 20, z: int = 500.

A.3.2 Compare with the running time of ESM-MSA vs HHblits

n_iter: int = 1, e_value: float = 0.0001, maxseq: int = 1000, realign_max: int = 100000, maxfilt: int = 100000, min_prefilter_hits: int = 1000, all_seqs: bool = False, p: 20, z: int = 500.

Table 8: HHblits speed compared with ESM-MSA (retieval & alignment). The values means how many MSAs are searched in 12h with 12-core CPU.

Model	Number of MSAs
HHblits	1509
ESM-MSA	**8800**

A.4 Contact map Results on SCOPe

See Table 9 for details.

A.5 Antibiotic Resistance dataset detail

The Antibiotic Resistance dataset is derived from experimentally verified bacterial antibiotic resistance proteins from the Comprehensive Antibiotic Resistance Database (CARD), and redundant sequences with 100% identity are removed using the CD-HIT tool [25]. Finally, a total of 2602 protein sequences from 19 antibiotic classes are constructed for functional classification and analysis.

A.6 GPU cluster

All our experiments are performed on the NVIDIA A40 with 48G GPU memory.

A.7 Remote Homology Detection (Evolutionary Understanding Task)

Here, we add the results of the remote homology detection task, which has exactly the same training and testing set in TAPE. As shown in Table 10, we can make the same observations as in the annotation prediction task. First, the performance of all three PLMs is substantially improved by pre-training, which shows the representation ability of them; Second, ESM-1b performs better than MSA-Transformer and Evoformer. Our results are consistent with those in TAPE. It is worth noting

that although the homology detection task measures a model's ability to detect structural similarity, it is essentially formulated as a protein-level annotation or classification task, like the MIB and ABR tasks in this paper. By comparison, the typical structural prediction tasks, including the SS prediction, contact prediction, and 3D structure prediction, are atom- or residue-level prediction, where residue is often represented by the C_α or C_β atom.

Table 10: Results of the remote homology detection task. 'scratch' means random initialization for model parameters.

Model	Pretrained	Scratch
ESM-1b	**0.31**	0.12
MSA-Transformer	0.22	**0.13**
Evoformer	0.23	0.11