# Fine-tuning protein language models boosts predictions across diverse tasks

**Robert Schmirler**[1,2,3 *], **Michael Heinzinger**[2,3], **and Burkhard Rost**[2,3,4,5]

[1]AbbVie Deutschland GmbH & Co. KG, Innovation Center, BTS IR LU, Ludwigshafen, 67061, Germany
[2]TUM (Technical University of Munich), Faculty of Informatics, Munich, 80333, Germany
[3]TUM, TUM School of Computation, Information and Technology (CIT), Garching, 85748, Germany
[4]TUM, Institute for Advanced Study (TUM-IAS), Garching, 85748, Germany
[5]TUM, TUM School of Life Sciences Weihenstephan (WZW), Freising, 85354, Germany
[*]corresponding author: robert.schmirler@tum.de

## ABSTRACT

Prediction methods inputting embeddings from protein Language Models (pLMs) have reached or even surpassed state-of-the-art (SOTA) performance on many protein prediction tasks. In natural language processing (NLP) fine-tuning Language Models has become the *de facto* standard. In contrast, most pLM-based protein predictions do not back-propagate to the pLM. Here, we compared the fine-tuning of three SOTA pLMs (ESM2, ProtT5, Ankh) on eight different tasks. Two results stood out. Firstly, task-specific supervised fine-tuning almost always improved downstream predictions. Secondly, parameter-efficient fine-tuning could reach similar improvements consuming substantially fewer resources. Put simply: always fine-tune pLMs and you will mostly gain. To help you, we provided easy-to-use notebooks for parameter efficient fine-tuning of ProtT5 for per-protein (pooling) and per-residue prediction tasks at https://github.com/agemagician/ProtTrans/tree/master/Fine-Tuning.

## 1 Introduction

Transformer-based[1] language models (LMs) have revolutionized Natural Language Processing (NLP). Large language models (LLMs) now perform at or above average human level. The newest generative models (GPT4[2] or PaLM2[3]) expand upon NLP through impressive capabilities in coding, math, and even common sense reasoning[4]. The success of LLMs has led to their wide-spread application from computer vision[5] over time series forecasting[6] to biologic language models[7–13].

For instance, trained on many protein sequences, so called protein language models (pLMs)[14–17] generate so called embeddings (numerical values extracted from the last hidden layers of pLMs) that succeed as exclusive input to subsequent supervised prediction tasks. Over the last 30 years, the *de facto* standard in protein prediction has been the use of evolutionary information[18]. PLM-based predictions reach such state-of-the-art (SOTA) expert devices for many prediction task, often even advancing the SOTA.

PLMs learn from large data sets without any experimental annotation other than the sequence. The information extracted by the pLM can be readily transferred to any protein-related prediction task. This generality makes pLMs suitable to a wide variety of prediction tasks spanning from secondary structure[14] over membrane regions[19], intrinsic disorder[20], and protein structure[16,21] to predictions of stability[22,23] or solubility[24]. Successful applications to more function-related predictions include the identification of paratopes[25], epitopes[26], and signal peptides[27], as well as, other tasks, e.g., related to the effect of sequence variation[28–31]). Embedding-based predictions seem particularly advantageous when experimental data are very limited[32].

These so-called embeddings are used as input for common machine learning methods like feed forward or convolutional neural networks (CNN). More complex architectures have also been explored[33]. Continued unsupervised training can focus models on specific protein families[34] or enrich embeddings with structural information essentially creating a bi-lingual pLM[35]. Training specialist models from scratch on smaller, specific proteins, e.g., antibodies[25,36], seems an alternative to "continued training" (train pLM on large generic data and refine on specific proteins).

Here, we evaluated the impact of task-specific supervised fine-tuning. We added a simple network as *prediction head* on top of the pLM encoder and applied supervised training to the model (pLM encoder and prediction head). We compared the results to predictions using pre-trained embeddings (training the prediction head using pre-computed embeddings). For the larger T5 based models (ProtT5[14], ProstT5[35] and both Ankh[15] models) we applied Low Rank Adaptation (LoRA)[37], a particular version of a more general approach known as PEFT (Parameter Efficient Fine-Tuning)[38]. Freezing most of the model and updating only a small fraction of the weights requires less hardware and prevents so called *catastrophic forgetting*[39,40]. This will become especially relevant as pLMs[17] grow larger. For pLMs, fine-tuning remains less studied than for NLP[38,41,42], although some

prediction tasks have been reported to profit from supervised, task-specific fine-tuning[43–45]. We assessed diverse prediction tasks from eight previously established benchmark datasets and three model architectures (pLMs) over a wide range of model sizes, aiming to draw a more general conclusion.

## 2 Results and Discussion

**Top-level comparison of models and tasks.** We trained 455 individual prediction methods (175 for fine-tuning, 280 using frozen embeddings from pre-trained pLMs - protein Language Models) comprising seven models (Table 3), each trained on eight different datasets (Table 2). Each model-task combination was optimized multiple times with different random seeds and we reported average values of those reruns. We used the corresponding validation set to select the best checkpoint for each training run. For each prediction task (PT), we compared the performance between fine-tuning and pre-training as:

$$\Delta(PT) = performance(PT)_{finetuned} - performance(PT)_{pretrained} \tag{1}$$

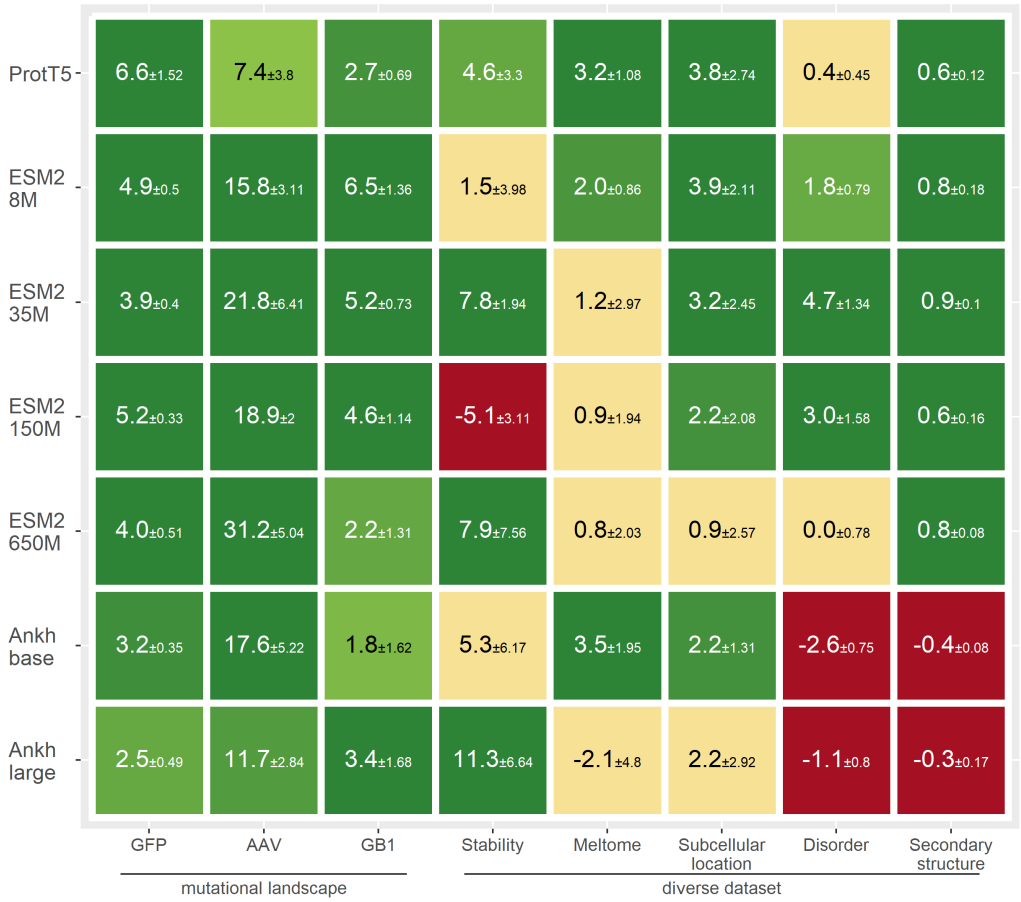


**Figure 1. Finetuning improved for most pLMs and tasks.** Values are percentage differences between the fine-tuned and pre-trained models (Eqn.1) for the eight prediction tasks (PTs on the x-axis). Due to the difference in the nature of the PTs, we had to use different performance measures, namely the Spearman rank correlation (GFP, AAV, GB1, stability, meltome and disorder), 10-class accuracy (Q10: sub-cellular location), and 3-class per-residue accuracy (Q3: secondary structure). Each tile compares the fine-tuning of one specific model against the pre-trained embedding-based prediction for one specific prediction task. Green tiles mark statistically significant performance increases (exceeding 1.96 standard errors; fine-tuning over pre-trained embeddings), yellow tiles mark statistically insignificant changes (0 lies within the error margins of ±1.96 stderr) and for red tiles supervised fine-tuning significantly decreased performance. Error estimates (±percentage values) represent the 95% confidence intervals (CI) from multiple runs of the same experiment (same model, identical data split) using different random seeds.

Even though not all improvements were significant, for the ProtT5-XL-U50[14] model (labelled *ProtT5*) and all four tested ESM2 models[16], supervised fine-tuning increased performance for all combinations of pLMs and tasks (Fig. 1, detailed results in supplementary online material (SOM) Tables S1-S6), with one exception (ESM2-150M / Stability). Both Ankh[15] models deviated from this result, gaining significantly by fine-tuning only for the mutational landscape data sets (GFP, AAV and GB1: green in Fig. 1).
A more detailed PT specific evaluation can be found in SOM 3. Based on those investigations, we emphasize the importance of high quality data splits. Validation sets without information leakage to train and test set that still capture test set behaviour well are hard to achieve. But those are absolutely necessary to prevent over-fitting and get the best predictors possible.

**Insignificant gain for secondary structure prediction.** For per-residue, three-class secondary structure prediction (helix, strand, other), fine-tuning improved only slightly (Fig. 2a; up to 1.2 percentage points for both data sets: CASP12[46] and NEW364[14]). We confirmed this for both the original general purpose ProtT5[14], and for the bilingual, structure-tuned ProstT5[35] model.
This might be explained by a combination of two effects: on the hand, secondary structure might already have been captured in unsupervised pre-training. On the other hand, today's performance may approach the upper-limit achievable[47].

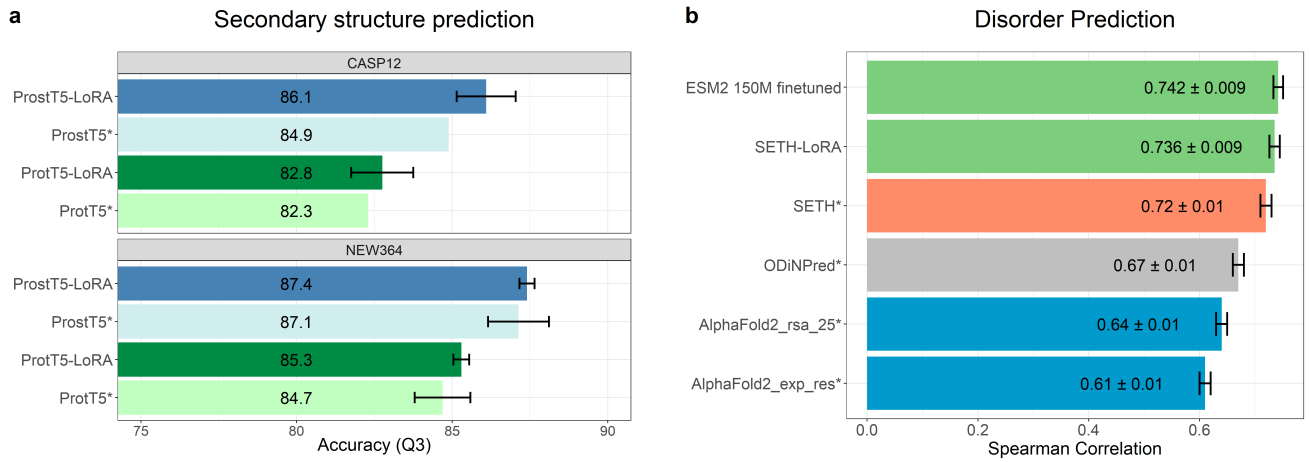


**Figure 2. Disorder improved while secondary structure is hardly affected. a:** Values for the pre-trained models (ProtT5[14] and ProstT5[35]) taken from literature[35] and marked by asterisk (*). We included two previously used data sets (CASP12[46] and NEW364[14]). Error bars mark the 95% confidence intervals, estimated by bootstrapping (values not available for pre-trained CASP12 results). **b:** Intrinsically disordered residues can be described by so-called CheZOD scores[48]. The x-axis shows the Spearman correlation between experimental and predicted CheZOD scores for six methods. Values marked by asterisks (*) taken from the literature[20]. Finetuning results in green, pLM-based without MSA (SETH[20]) in orange, MSA-based SOTA in gray[48,49], and MSA-based AlphaFold2[50] in blue. Error bars mark the 95% confidence intervals (CI), estimated via bootstrapping.

**Fine-tuning boosted disorder prediction.** SETH[20] reached the level of MSA-based SOTA methods, such as ODiNPred[48] in the prediction of per-residue protein disorder as described by CheZOD scores[48]. SETH inputs ProtT5 embeddings into a 2-layer CNN. Keeping those hyper-parameters and adding LoRA fine-tuning (dubbed SETH-LoRA ), improved performance by 2.2 percentage points (from Spearman 0.72 to 0.736, Fig. 2b). The fine-tuned much smaller 150M parameter ESM2 model bested (Spearman: 0.742) all solutions compared (Fig. 2b), including its larger counterparts, i.e., ESM2 with 650M parameters (SOM Table S4). Compared to SETH-LoRA that unfroze only 2.5 million of the 1.2 billion ProtT5 parameters, for ESM2 150M all parameters were finetuned. However, both fine-tuning approaches reached similar performance (Fig. 2b). While fine-tuning often led to improved performance (Fig. 1), selection of the absolute best model for a specific task remained less clear. For now we, therefore, recommend testing different pLMs for your task.

**Sub-cellular location: LoRA topped pooling.** Most predictions of sub-cellular location are based on input signals averaged over entire proteins. Embedding-based solutions do this through pooling, i.e., embeddings derived from averaging over all residue-level embeddings[14]. Light Attention (LA) can substantially improve over such averaging by learning the optimal per-residue signal and combining this with the average[33]. LoRA fine-tuning combined the advantage of a small model (simple neural network over convolutions) put on top of average-pooled last hidden states with the learned, weighted averaging of

LA. Thereby, LoRA fine-tuning numerically even surpassed LA, although the difference was statistically significant only at a confidence interval (CI) above 88%, not at the more commonly used CI=95% (Table 1).

**Table 1. Q10 sub-cellular localization prediction** * Values are taken from previous work[33]. Performance measured as 10-class accuracy (Q10) on the "set_HARD" test dataset.

| Model | set_HARD |
|---|---|
| ProtT5 FFN* | 61.3±1.0 |
| ProtT5 LA* | 65.2±0.6 |
| ProtT5 FFN-LoRA | **66.2±0.6** |

**Fine-tuning captures some effects of mutations.** We only found minor differences between the fine-tuned pLMs for the mutations landscape tasks (SOM Tables S4 and S6). Thus, we reported the average performance across all pLMs (Fig. 3). We compared these results to homology based inference (using MMseqs2[51] *search*) and Reference-free analysis (RFA)[52].
RFA fit a decent first-order model capturing the fitness landscape reflecting some of the possible mutations for *GB1* (protein G domain B1[53]; all possible variants at four positions, i.e., for four residues). For the *AAV2*[54] (adeno-associated virus 2) landscape where a 28-residue window was mutated, RFA performed less promisingly.
For *GFP* (green fluorescent protein[55]) the RFA analyses failed, because not all test set substitutions were present during training (i.e. some specific substitutions XnY, where amino acid X at position n is mutated to Y were never seen during training but occurred in the test set).
These results suggested that pLMs captured more relevant information than simple sequence similarity between sequences in training and testing sets. Explicit effect modeling (e.g. using RFA) failed for unseen substitutions, insertions, and deletions. In contrast, pLMs naturally manage all those cases.

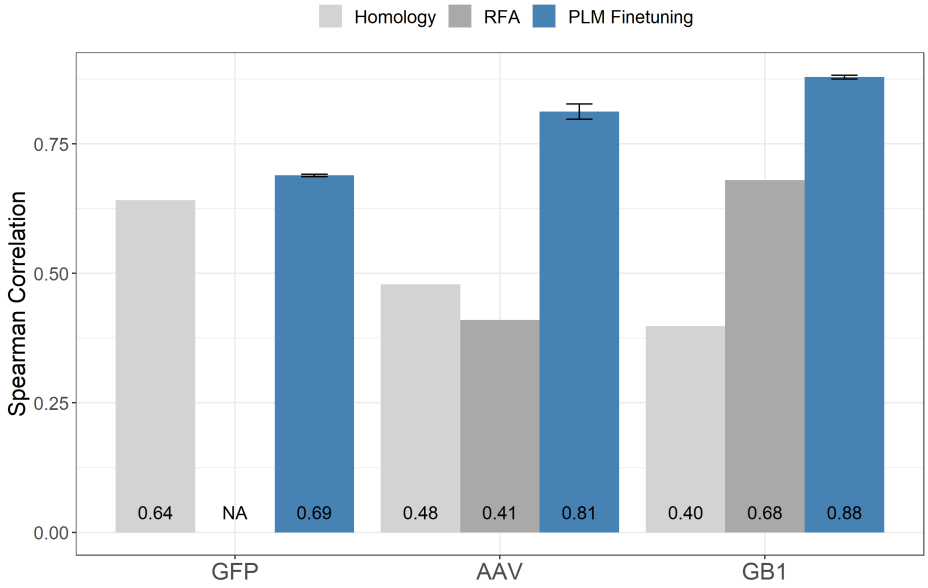


**Figure 3. Simple methods do not sufficiently capture mutational effect.** Blue bars: average performance across all fine-tuned pLMs (error bars: 95% CI) Gray bars: two simple baseline methods: *Homology*: MMseqs2[51] inferred the fitness value of test set proteins from most sequence-similar *homolog* in the training set. *RFA* (reference-free analysis[52]) fits models based on the assumption, that the most of the mutational effect can be described as the sum of low-order effects.

## 3 Conclusion

We applied fine-tuning[41,42] and PEFT[37,38] to several different prediction tasks (spanning from the per-protein level of sub-cellular location, and thermostability to the per-residue level of secondary structure and disorder) gauging success by comparing the performance of feeding embeddings from protein Language Models (pLMs: ProtT5[14], Ankh[15], ESM2[16]) into subsequent

supervised prediction methods with task specific supervised finetuning of the pLMs models. We found that fine-tuning can clearly improve performance. The extent of this improvement varied by task and pLM/model. LoRA fine-tuning of ProT5 seemed to have unlocked additional degrees of freedom within the model. For per-protein predictions of location, the LoRA optimization may have learned weighted pooling of the last hidden layer, and that improves significantly over average pooling[33]. More generally, the last hidden layer has been optimized for the un-supervised pre-training objective (learning to reproduce masked sequences). This optimization might not be best for any downstream task[56,57]. Fine-tuning may exactly address this issue by optimizing the models output for particular tasks. Lastly, the transformer models might extract additional information directly from the task specific training. Results from randomly initialized smaller ESM2 models supported this view (SOM Figure S3, Table S13).

Overall, our results revealed the gains initially observed in NLP from supervised task-specific fine-tuning of LLMs[41,42] to also apply to large protein LMs (pLMs). We therefore suggest to add supervised finetuning as an additional optimization step when training pLM based predictors to see how much you can improve for your specific task.

# 4 Methods

## 4.1 Data

The data sets differentiated two aspects: *prediction task level* and *sequence diversity* (Table 2). The *prediction task level* distinguished per-residue (e.g. secondary structure) and per-protein (e.g. sub-cellular location). *Sequence diversity* distinguished between data sets with many sequence-diverse proteins and those from mutational studies analyzing single protein through deep mutational scanning experiments[58].

**Table 2. Task specific datasets**

| Prediction Level | Sequence Diversity | Dataset | # of Sequences | | | Average Seq. Length |
|---|---|---|---|---|---|---|
| | | | Train | Validation | Test | |
| per Protein | Mutational Landscapes | GFP[22,59] | 21,446 | 5,362 | 27,217 | 237.0 |
| | | AAV[23,60] | 28,626 | 3,181 | 50,776 | 736.3 |
| | | GB1[23,61] | 2,691 | 299 | 5,743 | 265.0 |
| | Diverse Datasets | Stability[22,62] | 53,614 | 2,512 | 12,851 | 45.0 |
| | | Meltome[23,63] | 22,335 | 2,482 | 3,134 | 544.5 |
| | | Sub. Loc.[33,64] | 9,503 | 1,678 | 490 | 519.9 |
| per Residue | | Disorder[20,48] | 1,056 | 118 | 117 | 118.1 |
| | | Sec. Str.[14,65] | 9,712 | 1,080 | 364 | 255.0 |

**Mutational landscapes:** These data comprised fitness landscapes for the green fluorescent protein (GFP[55]), the adeno-associated virus 2 capsid protein VP-1 (AAV2[54]) and the first binding domain of the Protein G (GB1[53]). All three constitute regression tasks that measure prediction performance by ranking the correlation between the predicted and the experimentally measured property for each set. For *GFP* the property/fitness was measured through fluorescence intensity (experimental data: Sarkisyan et al.[59] split of data: Rao et al.[22]). Training and validation set sequences were all within Hamming distance 3 (i.e. all variants up to three changes from wild type) For the *AAV* task, fitness was measured as the viability for packaging DNA payloads. The authors[60] mutated a 28-amino acid window to create the original data. We used the "2-vs-rest" data split from the FLIP benchmark[23] in which $\leq 2$ are in the training and validation sets, while all variants with more mutations are grouped into the test set. The *GB1* fitness score measures stability and binding affinity. The original experiment[61] mutated four positions we took the „three-vs-rest" data split from FLIP.

**Per-protein predictions:** This subgroup also included three prediction tasks. The first two sets focused on stability prediction formulated as regression tasks in analogy to the fitness landscapes described above. Stability predictions were assessed on measurements of protease susceptibility for *de novo* designed mini-proteins[62]. We reused the TAPE data split[22], where training and validation sets consist of sequences from four design cycles while the test set holds seventeen 1-Hamming distance neighborhoods to promising candidates. *Meltome* utilizes data from measuring thermostability for proteins from 13 species[63]. We used the "mixed" split from the FLIP benchmark[23], which uses MMseqs2[51] to create clusters with >20% pairwise sequence identity. Excluding any pair within the same cluster from test and train sets, reduces the odds of information leakage. The third data set included DeepLoc[64] data (incl. training/validation/testing splits) along with a novel test data set (*setHARD*) introduced

recently[33]. The task is to predict sub-cellular location in one of ten classes. MMseqs2[51] removed all sequences with >20% pairwise sequence identity between training, validation, and test sets.

**Per-residue predictions:** *Disorder predictions* used CheZOD data[48] from nuclear magnetic resonance (NMR) spectroscopy. We used previously published data splits[20] applying MMseqs2[51] clustering at <20% pairwise sequence identity (no pair of proteins between training and testing was more similar). The task is to predict CheZOD scores[49] which resemble a continuous scale to quantify the level of intrinsice disorder for each residue in a protein. We benchmarked *secondary structure predictions* through data sets provided by *NetSurfP-2.0*[65] distinguishing three classes (H: helix, E: strand, and C: other/non-regular). We used a more recent split of these data adhering to more stringent redundancy-reduction, along with another recent test set (*NEW364*)[14].

**Performance measurement:** To ease comparisons, we measured performance in the same manner the data set developers did (Table 2). All regression tasks were evaluated using Spearman rank correlations, i.e., all three mutational landscapes (GFP, AAV, GB1), both stability related data sets (Stability and Meltome), as well as, the per-residue regression of Disorder. For the classification tasks, accuracy was defined as a 10-class per-protein accuracy (Q10) for sub-cellular location, and a 3-class per-residue accuracy (Q3) for secondary structure.

### 4.2 Pre-trained pLMs

The pLMs used differed in size and architecture (Table 3), ranging from 8 million (ESM2 8M) to 3 billion parameters (ESM2 3B). ESM2 pLMs are RoBERTa[66] based encoder models trained using a unsupervised masked language modeling objective[16]. The other three pLMs are built on T5[67], an encoder-decoder LLM pre-trained applying span masking[14,15]. We initialized our models using the pre-trained checkpoints available on Huggingface.

**Table 3. Utilized protein language models**

| Model | Architecture (Pretraining) | Number of Parameters (Encoder) | Encoder Layers | Emb. Size | Huggingface Model Checkpoint |
|---|---|---|---|---|---|
| Ankh Base | Enc-Dec | 736 M | 48 | 768 | ankh-base |
| Ankh Large | | 1.9 B | 48 | 1536 | ankh-large |
| ProtT5 | | 1.2 B | 24 | 1024 | prot_t5_xl_uniref50 |
| ProstT5 | | 1.2 B | 24 | 1024 | ProstT5 |
| ESM2 8M | Enc | 8 M | 6 | 320 | esm2_t6_8M_UR50D |
| ESM2 35M | | 35 M | 12 | 480 | esm2_t12_35M_UR50D |
| ESM2 150M | | 150 M | 30 | 640 | esm2_t30_150M_UR50D |
| ESM2 650M | | 650 M | 33 | 1280 | esm2_t33_650M_UR50D |
| ESM2 3B | | 3.0 B | 36 | 2560 | esm2_t36_3B_UR50D |

### 4.3 Model training

**Top-level comparison:** We compared pre-trained to fine-tuned models as follows: For the pre-trained results, we generated embeddings for all data sets. For per-protein tasks, we averaged over the sequence length yielding a *1 x embedding_size*-dimensional vector for each protein. For per-residue tasks, we used all residue embeddings and their respective labels. This resulted in vectors of the same dimension as for the per-protein task (*1 x embedding_size*), albeit this time for each residue, i.e., each amino acid position. Next, we trained a single fully connected layer of size 32, inputting exclusively these embeddings and outputting either a single value (regression) or going into another output layer with one neuron for each possible output class, followed by a softmax layer to get a probability distribution. Reaching a plateau in training loss ended training. Each training was repeated five times with different random seeds.

For fine-tuning, we added the same fully connected layer (size 32) on top of the language model encoder as a prediction head. For ProtT5[14] and Ankh[15], we also used average pooling of the last hidden states over the sequence length dimension during training on per protein tasks. ESM2[16] suggests connecting the prediction head only to the very first token of the sequence (special token "<CLS>") which we did. Training was repeated three times with different random seeds. Training ended when training and validation loss flattened out. Due to hardware constraints we applied Parameter Efficient Fine-Tuning (PEFT[38]) to ProtT5 and Ankh. In particular, we applied the Low-Rank Adaptation (LoRA[37]) implementation of PEFT. For the ESM2 models, we fine-tuned all model weights.

For both, embedding and fine-tuning we reported the performance of the checkpoint with the lowest validation loss for each experiment. We also did a limited hyperparameter optimization at the beginning. Once selected, the hyperparameters were left unchanged for most of the comparisons (for details: SOM Tables S7 and S8). We used the Adam optimizer[68] with default parameters. To realize the batch size for model fine-tuning we applied gradient accumulation as needed with the given hardware. All training ran on a single NVIDIA A10G GPU with 24GB.

**Per-residue secondary structure:** We fine-tuned five models for ProtT5[14] and another five for ProstT5[35], initializing with different random seeds. For ProstT5, we added the prefix token "<fold2aa>" to each sequence, to make the model aware of the input type (amino acid rather than structure because ProstT5 is the only existing bilingual pLM). We trained for five epochs and calculated the validation loss at the end. For both models (ProtT5 and ProstT5), utilizing the same two-layer CNN which was applied previously[14,35]for predictions based on raw embeddings, kept results comparable. Of the five, we selected the model with lowest validation loss and measured performance on the previously established CASP12[46] and NEW364[14] data sets. We established confidence intervals through bootstrapping (Fig. 2a).

**Per-residue disorder:** We fine-tuned two model variants to compare to other methods[20]. *SETH-LoRA* was based on ProtT5 using the same two-layer CNN as the original SETH. *ESM2 150M* reused the ESM2 setup from the top level evaluation here (last hidden states of "<CLS>" token with single dense layer prediction head). For both variants, we trained five models with different random seeds for ten epochs, and calculated the validation loss twice per epoch, selecting the model with lowest validation loss out of the 100 checkpoints (5 random seeds, 10 epochs, 2 validations per epoch). Bootstrapping provided confidence intervals (Fig. 2b).

**Per-protein location:** For the 10-class classification, we reused the single layer dense network from our top level evaluation. We trained five models with different random seeds for five epochs. Validation loss was calculated twice per epoch. From each run we selected the model with the lowest validation loss. We then calculated Q10 accuracy and standard deviation from all five selected models on the "set_hard"[33] and report the average values (Table 1).

**Fitness landscapes:** We compared fine-tuned pLMs with two baselines, namely, homology based inference (HBI) and Reference-free analysis (RFA)[52]. We averaged test performance over all fine-tuned pLMs (selected checkpoints with lowest loss on validation set) for the three mutational landscape data sets (GFP, AAV, GB1).
**HBI**: We searched with *MMSeqs2*[51] using the test set as query and the training set as target. For each test set entry, we assigned the fitness value of the top hit (*closest homolog*) in the training set.
**RFA**: The RFA R implementation could be applied to the GB1 data without any adaptation. For AAV we had to remove all sequences with lengths $\neq 735$ (algorithm fails on inserts and deletions). This reduced the training data by about 1%. RFA failed for GFP because some substitutions in the test set were missing from the training set. For AAV and GB1, we fitted a first and a second-order model and reported results for the better of the two.

## Data availability

All data sets analyzed are freely available through the original sources:
GFP and stability: https://github.com/songlab-cal/tape
AAV, GB1, Meltome and secondary structure: https://github.com/J-SNACKKB/FLIP
Location: https://github.com/HannesStark/protein-localization
Disorder: https://github.com/DagmarIlz/SETH

For ease of access, we packaged all those data sets and made those available through: https://github.com/RSchmirler/data-repo_plm-finetune-eval/tree/main. When using those data, please quote and consult the authors of the original data sets.
All data generated during this study as well as data used to generate figures and tables is also available in this repository.

The Reference-free analysis (RFA) R scripts were downloaded from https://github.com/whatdoidohaha/RFA/tree/main.
For finetunig ESM2 models we used the notebooks available at https://colab.research.google.com/github/huggingface/notebooks/blob/main/examples/protein_language_modeling.ipynb with minor modifications to the data pipeline and training parameters (refer to Methods).
Notebooks to LoRA finetune T5 based pLMs are made available by us at https://github.com/agemagician/ProtTrans/tree/master/Fine-Tuning. Those notebooks are configured to use ProtT5 but can easily be adapted to other T5 based models like Ankh or ProstT5.

## References


1. Vaswani, A. *et al.* Attention is all you need. *Adv. neural information processing systems* **30** (2017).
2. OpenAI. Gpt-4 technical report. Preprint at https://arxiv.org/abs/2303.08774 (2023).
3. Anil, R. *et al.* Palm 2 technical report. Preprint at https://arxiv.org/abs/2305.10403 (2023).
4. Bubeck, S. *et al.* Sparks of artificial general intelligence: Early experiments with gpt-4. Preprint at https://arxiv.org/abs/2303.12712 (2023).
5. Liu, Z. *et al.* Swin transformer: Hierarchical vision transformer using shifted windows. *Proc. IEEE/CVF international conference on computer vision* 10012–10022 (2021).
6. Zhou, H. *et al.* Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proc. AAAI conference on artificial intelligence* **35**, 11106–11115 (2021).
7. Heinzinger, M. *et al.* Modeling aspects of the language of life through transfer-learning protein sequences. *BMC bioinformatics* **20**, 1–17 (2019).
8. Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M. & Church, G. M. Unified rational protein engineering with sequence-based deep representation learning. *Nat. methods* **16**, 1315–1322 (2019).
9. Bepler, T. & Berger, B. Learning protein sequence embeddings using information from structure. Preprint at https://arxiv.org/abs/1902.08661 (2019).
10. Rives, A. *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl. Acad. Sci.* **118**, e2016239118 (2021).
11. Trotter, M. V., Nguyen, C. Q., Young, S., Woodruff, R. T. & Branson, K. M. Epigenomic language models powered by cerebras. Preprint at https://arxiv.org/abs/2112.07571 (2021).
12. Dalla-Torre, H. *et al.* The nucleotide transformer: Building and evaluating robust foundation models for human genomics. Preprint at https://www.biorxiv.org/content/10.1101/2023.01.11.523679v3 (2023).
13. Yang, F. *et al.* scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nat. Mach. Intell.* **4**, 852–866, DOI: 10.1038/s42256-022-00534-z (2022).
14. Elnaggar, A. *et al.* Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis Mach. Intell.* **44**, 7112–7127, DOI: 10.1109/TPAMI.2021.3095381 (2022).
15. Elnaggar, A. *et al.* Ankh: Optimized protein language model unlocks general-purpose modelling. Preprint at https://arxiv.org/abs/2301.06568 (2023).
16. Lin, Z. *et al.* Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* **379**, 1123–1130 (2023).
17. Chen, B. *et al.* xtrimopglm: Unified 100b-scale pre-trained transformer for deciphering the language of protein. Preprint at https://www.biorxiv.org/content/10.1101/2023.07.05.547496v3 (2023).
18. Rost, B. & Sander, C. Prediction of protein secondary structure at better than 70% accuracy. *J. molecular biology* **232**, 584–599 (1993).
19. Bernhofer, M. & Rost, B. Tmbed: transmembrane proteins predicted through language model embeddings. *BMC bioinformatics* **23**, 326 (2022).
20. Ilzhöfer, D., Heinzinger, M. & Rost, B. Seth predicts nuances of residue disorder from protein embeddings. *Front. Bioinforma.* **2** (2022).
21. Weißenow, K., Heinzinger, M. & Rost, B. Protein language-model embeddings for fast, accurate, and alignment-free protein structure prediction. *Structure* **30**, 1169–1177 (2022).
22. Rao, R. *et al.* Evaluating protein transfer learning with tape. *Adv. neural information processing systems* **32**, 9689–9701 (2019).
23. Dallago, C. *et al.* Flip: Benchmark tasks in fitness landscape inference for proteins. Preprint at https://www.biorxiv.org/content/10.1101/2021.11.09.467890v2 (2021).
24. Feng, J., Jiang, M., Shih, J. & Chai, Q. Antibody apparent solubility prediction from sequence by transfer learning. *Iscience* **25** (2022).
25. Leem, J., Mitchell, L. S., Farmery, J. H., Barton, J. & Galson, J. D. Deciphering the language of antibodies using self-supervised learning. *Patterns* **3** (2022).

**26.** Clifford, J. N. *et al.* Bepipred-3.0: Improved b-cell epitope prediction using protein language models. *Protein Sci.* **31**, e4497 (2022).

**27.** Teufel, F. *et al.* Signalp 6.0 predicts all five types of signal peptides using protein language models. *Nat. biotechnology* **40**, 1023–1025 (2022).

**28.** Groth, P. M., Michael, R., Salomon, J., Tian, P. & Boomsma, W. Flop: Tasks for fitness landscapes of protein wildtypes. Preprint at https://www.biorxiv.org/content/10.1101/2023.06.21.545880v1 (2023).

**29.** Marquet, C. *et al.* Embeddings from protein language models predict conservation and variant effects. *Hum. Genet.* **141**, 1629–1647, DOI: 10.1007/s00439-021-02411-y (2022).

**30.** Nijkamp, E., Ruffolo, J., Weinstein, E. N., Naik, N. & Madani, A. Progen2: exploring the boundaries of protein language models. Preprint at https://arxiv.org/abs/2206.13517 (2022).

**31.** Notin, P. *et al.* Tranception: protein fitness prediction with autoregressive transformers and inference-time retrieval. *Int. Conf. on Mach. Learn.* 16990–17017 (2022).

**32.** Littmann, M., Heinzinger, M., Dallago, C., Weissenow, K. & Rost, B. Protein embeddings and deep learning predict binding residues for various ligand classes. *Sci. Reports* **11**, 23916 (2021).

**33.** Stärk, H., Dallago, C., Heinzinger, M. & Rost, B. Light attention predicts protein location from the language of life. *Bioinforma. Adv.* **1**, vbab035, DOI: 10.1093/bioadv/vbab035 (2021).

**34.** Madani, A. *et al.* Large language models generate functional protein sequences across diverse families. *Nat. Biotechnol.* 1–8 (2023).

**35.** Heinzinger, M. *et al.* Prostt5: Bilingual language model for protein sequence and structure. Preprint at https://www.biorxiv.org/content/10.1101/2023.07.23.550085v1 (2023).

**36.** Olsen, T. H., Moal, I. H. & Deane, C. M. Ablang: an antibody language model for completing antibody sequences. *Bioinforma. Adv.* **2**, vbac046 (2022).

**37.** Hu, E. J. *et al.* Lora: Low-rank adaptation of large language models. Preprint at https://arxiv.org/abs/2106.09685 (2021).

**38.** Ding, N. *et al.* Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nat. Mach. Intell.* **5**, 220–235 (2023).

**39.** French, R. M. Catastrophic forgetting in connectionist networks. *Trends cognitive sciences* **3**, 128–135 (1999).

**40.** Kirkpatrick, J. *et al.* Overcoming catastrophic forgetting in neural networks. *Proc. national academy sciences* **114**, 3521–3526 (2017).

**41.** Howard, J. & Ruder, S. Universal language model fine-tuning for text classification. *In Proc. 56th Annu. Meet. Assoc. for Comput. Linguist.* 328–339, DOI: 10.18653/v1/P18-1031 (2018).

**42.** Wortsman, M. *et al.* Robust fine-tuning of zero-shot models. *Proc. IEEE/CVF Conf. on Comput. Vis. Pattern Recognit.* 7959–7971 (2022).

**43.** Thumuluri, V. *et al.* Netsolp: predicting protein solubility in escherichia coli using language models. *Bioinformatics* **38**, 941–946 (2022).

**44.** Wang, D., Fei, Y. E. & Zhou, H. On pre-training language model for antibody. *The Eleventh Int. Conf. on Learn. Represent.* (2022).

**45.** Dumitrescu, A. *et al.* Tsignal: a transformer model for signal peptide prediction. *Bioinformatics* **39**, i347–i356 (2023).

**46.** Abriata, L. A., Tamò, G. E., Monastyrskyy, B., Kryshtafovych, A. & Dal Peraro, M. Assessment of hard target modeling in casp12 reveals an emerging role of alignment-based contact prediction methods. *Proteins: Struct. Funct. Bioinforma.* **86**, 97–112 (2018).

**47.** Rost, B., Sander, C. & Schneider, R. Redefining the goals of protein secondary structure prediction. *J. molecular biology* **235**, 13–26 (1994).

**48.** Dass, R., Mulder, F. A. & Nielsen, J. T. Odinpred: comprehensive prediction of protein order and disorder. *Sci. Reports* **10**, 14780 (2020).

**49.** Nielsen, J. T. & Mulder, F. A. Quantitative protein disorder assessment using nmr chemical shifts. *Intrinsically Disord. proteins: methods protocols* 303–317 (2020).

**50.** Jumper, J. *et al.* Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).

**51.** Steinegger, M. & Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. biotechnology* **35**, 1026–1028 (2017).

**52.** Park, Y., Metzger, B. P. & Thornton, J. W. The simplicity of protein sequence-function relationships. Preprint at https://www.biorxiv.org/content/10.1101/2023.09.02.556057v1 (2023).

**53.** Sauer-Eriksson, A. E., Kleywegt, G. J., Uhlén, M. & Jones, T. A. Crystal structure of the c2 fragment of streptococcal protein g in complex with the fc domain of human igg. *Structure* **3**, 265–278 (1995).

**54.** Russell, S. *et al.* Efficacy and safety of voretigene neparvovec (aav2-hrpe65v2) in patients with rpe65-mediated inherited retinal dystrophy: a randomised, controlled, open-label, phase 3 trial. *The Lancet* **390**, 849–860 (2017).

**55.** Chalfie, M., Tu, Y., Euskirchen, G., Ward, W. W. & Prasher, D. C. Green fluorescent protein as a marker for gene expression. *Science* **263**, 802–805 (1994).

**56.** Li, F.-Z., Amini, A. P., Yang, K. K. & Lu, A. X. Pretrained protein language model transfer learning: is the final layer representation what we want. *Proc. Mach. Learn. for Struct. Biol. Work. NeurIPS 2022* (2022).

**57.** Valeriani, L. *et al.* The geometry of hidden representations of large transformer models. Preprint at https://arxiv.org/abs/2302.00294 (2023).

**58.** Fowler, D. M. & Fields, S. Deep mutational scanning: a new style of protein science. *Nat. methods* **11**, 801–807 (2014).

**59.** Sarkisyan, K. S. *et al.* Local fitness landscape of the green fluorescent protein. *Nature* **533**, 397–401 (2016).

**60.** Bryant, D. H. *et al.* Deep diversification of an aav capsid protein by machine learning. *Nat. Biotechnol.* **39**, 691–696 (2021).

**61.** Wu, N. C., Dai, L., Olson, C. A., Lloyd-Smith, J. O. & Sun, R. Adaptation in protein fitness landscapes is facilitated by indirect paths. *Elife* **5**, e16965 (2016).

**62.** Rocklin, G. J. *et al.* Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science* **357**, 168–175 (2017).

**63.** Jarzab, A. *et al.* Meltome atlas—thermal proteome stability across the tree of life. *Nat. methods* **17**, 495–503 (2020).

**64.** Almagro Armenteros, J. J., Sønderby, C. K., Sønderby, S. K., Nielsen, H. & Winther, O. Deeploc: prediction of protein subcellular localization using deep learning. *Bioinformatics* **33**, 3387–3395, DOI: 10.1093/bioinformatics/btx548 (2017).

**65.** Klausen, M. S. *et al.* Netsurfp-2.0: Improved prediction of protein structural features by integrated deep learning. *Proteins: Struct. Funct. Bioinforma.* **87**, 520–527 (2019).

**66.** Liu, Y. *et al.* Roberta: A robustly optimized bert pretraining approach. Preprint at https://arxiv.org/abs/1907.11692 (2019).

**67.** Raffel, C. *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. *The J. Mach. Learn. Res.* **21**, 5485–5551 (2020).

**68.** Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. Preprint at https://arxiv.org/abs/1412.6980 (2014).


## Acknowledgements


RS is an employee of AbbVie. The design, study conduct, and financial support for this research were provided by AbbVie. AbbVie participated in the interpretation of data, review, and approval of the publication. All computational resources for this work were provided by AbbVie. MH and BR were supported by the Bavarian Ministry of Education through funding to the TUM, by a grant from the Alexander von Humboldt Foundation through the German Ministry for Research and Education (BMBF: Bundesministerium für Bildung und Forschung), and by a grant from Deutsche Forschungsgemeinschaft (DFG-GZ: RO1320/4-1).


## Author contributions statement

All authors contributed to the conception of this study. R.S. performed all experimental work, results analysis and creation of figures. The first draft of the manuscript was written by R.S. and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

## Additional information

No competing interests have to be disclosed.