LifeLU - UNDERSTANDING THE LANGUAGE OF LIFE

READING GROUP

Presenter: Özlem Şimşek

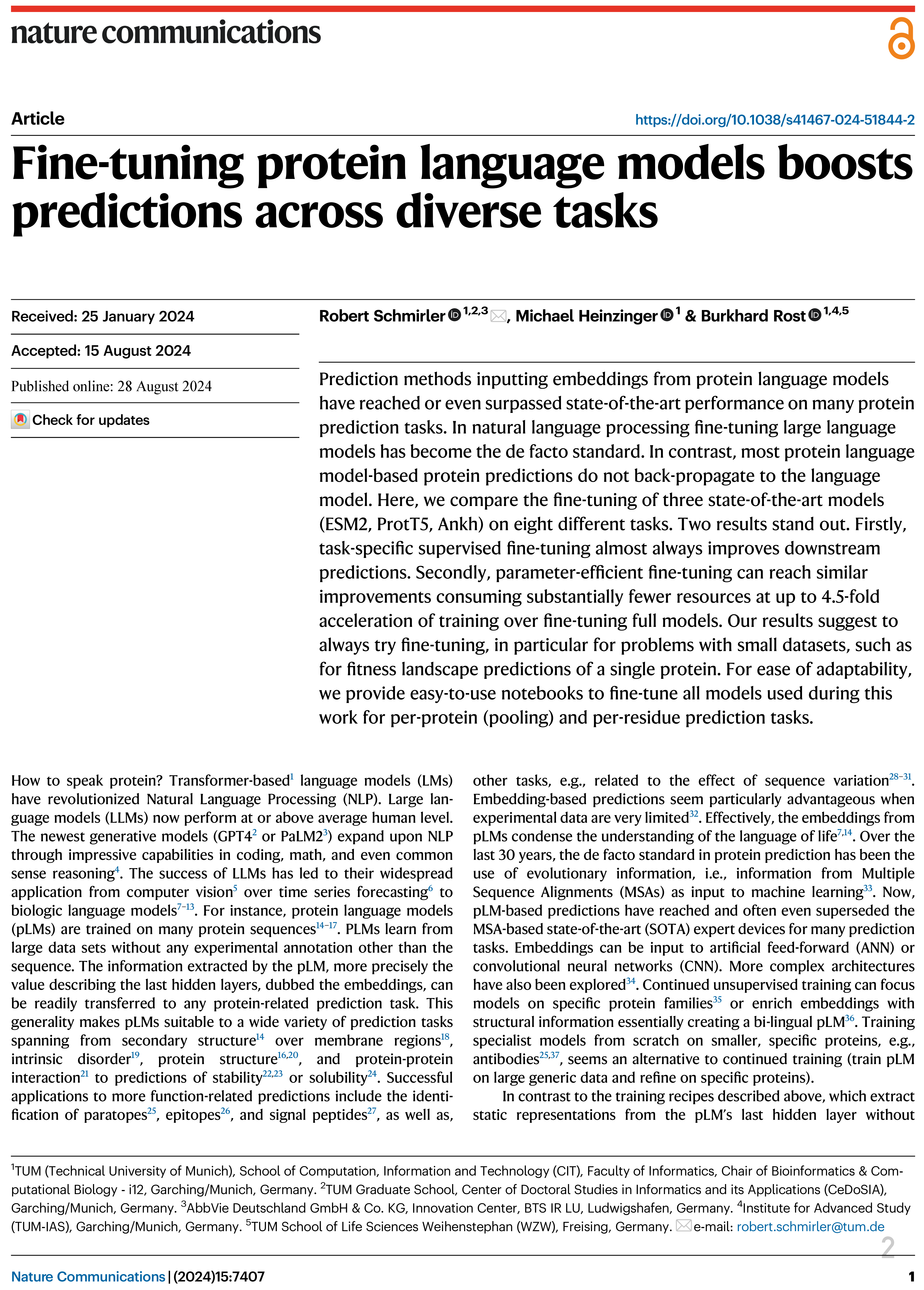
Paper: Fine-tuning protein language models boosts predictions across diverse tasks

Link: https://www.nature.com/articles/s41467-024-51844-2

Date: 20/03/2025

Fine-tuning protein language models boosts predictions across diverse tasks

Schmirler, R., Heinzinger, M., & Rost, B. (2024). Fine-tuning protein language models boosts predictions across diverse tasks. Nature Communications, 15, 2024



nature communications

Article

https://doi.org/10.1038/s41467-024-51844-2

Fine-tuning protein language models boosts predictions across diverse tasks

Received: 25 January 2024

Accepted: 15 August 2024

Published online: 28 August 2024

Check for updates

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

Prediction methods inputting embeddings from protein language models have reached or even surpassed state-of-the-art performance on many protein prediction tasks. In natural language processing fine-tuning large language models has become the de facto standard. In contrast, most protein language model-based protein predictions do not back-propagate to the language model. Here, we compare the fine-tuning of three state-of-the-art models (ESM2, ProtT5, Ankh) on eight different tasks. Two results stand out. Firstly, task-specific supervised fine-tuning almost always improves downstream predictions. Secondly, parameter-efficient fine-tuning can reach similar improvements consuming substantially fewer resources at up to 4.5-fold acceleration of training over fine-tuning full models. Our results suggest to always try fine-tuning, in particular for problems with small datasets, such as for fitness landscape predictions of a single protein. For ease of adaptability, we provide easy-to-use notebooks to fine-tune all models used during this work for per-protein (pooling) and per-residue prediction tasks.

How to speak protein? 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 widespread application from computer vision[5] over time series forecasting[6] to biologic language models[7–13]. For instance, protein language models (pLMs) are trained on many protein sequences[14–17]. PLMs learn from large data sets without any experimental annotation other than the sequence. The information extracted by the pLM, more precisely the value describing the last hidden layers, dubbed the embeddings, 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[18], intrinsic disorder[19], protein structure[16,20], and protein-protein interaction[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]. Effectively, the embeddings from pLMs condense the understanding of the language of life[7,14]. Over the last 30 years, the de facto standard in protein prediction has been the use of evolutionary information, i.e., information from Multiple Sequence Alignments (MSAs) as input to machine learning[33]. Now, pLM-based predictions have reached and often even superseded the MSA-based state-of-the-art (SOTA) expert devices for many prediction tasks. Embeddings can be input to artificial feed-forward (ANN) or convolutional neural networks (CNN). More complex architectures have also been explored[34]. Continued unsupervised training can focus models on specific protein families[35] or enrich embeddings with structural information essentially creating a bi-lingual pLM[36]. Training specialist models from scratch on smaller, specific proteins, e.g., antibodies[25,37], seems an alternative to continued training (train pLM on large generic data and refine on specific proteins).

In contrast to the training recipes described above, which extract static representations from the pLM's last hidden layer without

[1]TUM (Technical University of Munich), School of Computation, Information and Technology (CIT), Faculty of Informatics, Chair of Bioinformatics & Computational Biology - i12, Garching/Munich, Germany. [2]TUM Graduate School, Center of Doctoral Studies in Informatics and its Applications (CeDoSIA), Garching/Munich, Germany. [3]AbbVie Deutschland GmbH & Co. KG, Innovation Center, BTS IR LU, Ludwigshafen, Germany. [4]Institute for Advanced Study (TUM-IAS), Garching/Munich, Germany. [5]TUM School of Life Sciences Weihenstephan (WZW), Freising, Germany. e-mail: robert.schmirler@tum.de

SUMMARY

Aim: compare the fine-tuning of three state-of-the-art models on eight different tasks

Models:

- ESM2
- ProtT5
- Ankh

Tasks:

- GFP
- AAV
- GB1
- Stability
- Meltome
- Subcellular location
- Disorder
- Secondary structure

INTRODUCTION-1

- Before, there are MSA(multiple sequence alignment)-based methods.
- Now, there are pLM-based predictions.

- pLMs learn from large data sets without any experimental annotation other than the sequence.
- The embeddings from pLMs are transferred to many protein-related prediction tasks.

- Embedding-based predictions are advantageous when experimental data are very limited.

INTRODUCTION-2

- In most cases, the embeddings are used in downstream tasks as static representations from the pLM's last hidden layer without changing its weights.

- Here, we evaluated the impact of task-specific supervised fine-tuning by:
    - adding a simple ANN as a prediction head on top of the pLM encoder and
    - applying supervised training to both the pLM encoder and the prediction head

- For the larger models (ProtT5, ProstT5, Ankh models and ESM2B 3M), we utilized Low Rank Adaptation (LoRA) to accelerate training.
    - freezing most of the model and updating only a small fraction of the weights while fine-tuning

GLOSSARY

Tasks:

FLIP (Functional Landscape of Interacting Proteins) benchmark datasets *

Aim: predict protein fitness under constrained conditions

* C. Dallago, J. Mou, K. E. Johnston, B. Wittmann, N. Bhattacharya, S. Goldman, A. Madani, and K. K. Yang, “Flip: Benchmark tasks in fitness landscape inference for proteins,” *bioRxiv*, 2021 (2021 NeurIPS Benchmark track)

GLOSSARY

Parameter Efficient Fine-Tuning (PEFT)*:

- A method of improving the performance of pretrained LLMs and neural netwoeks for specific tasks or data sets.
- Trains a small set of parameters and preserves most of the large pretrained model's structure
- PEFT saves time and computational resources.

Low Rank Adaptation (LoRA):

- Introduced in 2021
- Uses twin low-rank decomposition matrices to minimize model weights and reduce the subset of trainable parameters even further.

* https://www.ibm.com/think/topics/parameter-efficient-fine-tuning

DATABASES-1

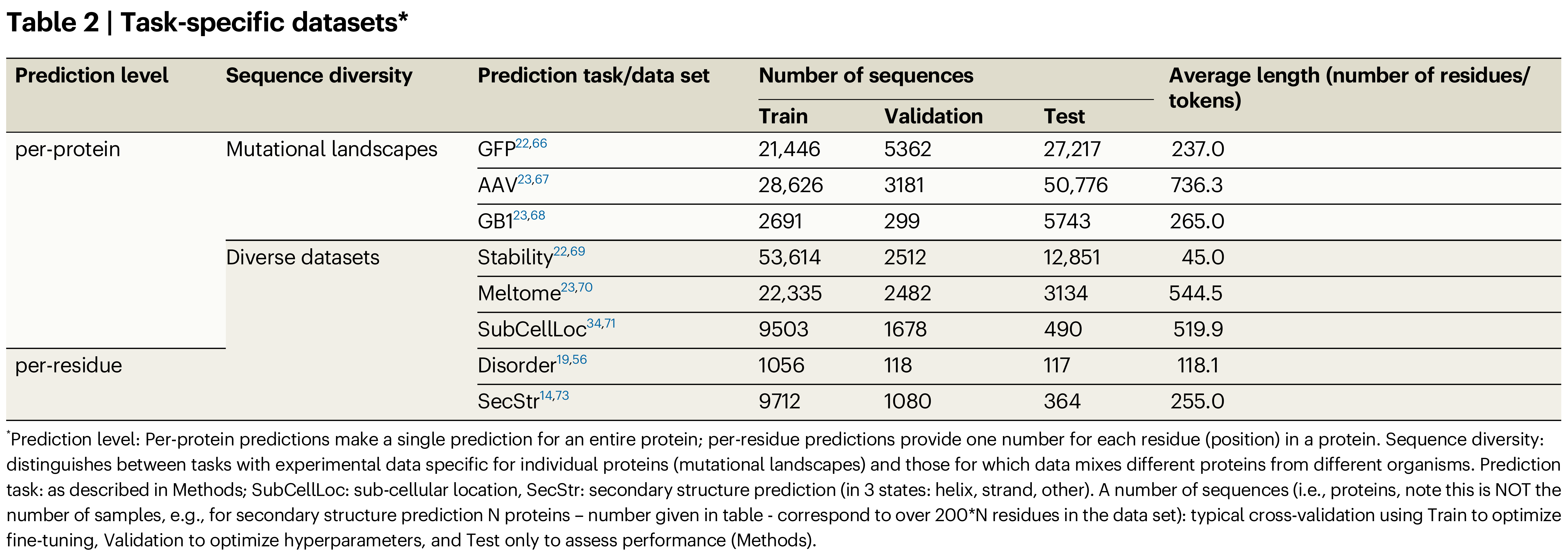


Table 2 | Task-specific datasets*

Prediction level	Sequence diversity	Prediction task/data set	Number of sequences			Average length (number of residues/ tokens)
			Train	Validation	Test	
per-protein	Mutational landscapes	GFP[22,66]	21,446	5362	27,217	237.0
		AAV[23,67]	28,626	3181	50,776	736.3
		GB1[23,68]	2691	299	5743	265.0
	Diverse datasets	Stability[22,69]	53,614	2512	12,851	45.0
		Meltome[23,70]	22,335	2482	3134	544.5
		SubCellLoc[34,71]	9503	1678	490	519.9
per-residue		Disorder[19,56]	1056	118	117	118.1
		SecStr[14,73]	9712	1080	364	255.0

*Prediction level: Per-protein predictions make a single prediction for an entire protein; per-residue predictions provide one number for each residue (position) in a protein. Sequence diversity: distinguishes between tasks with experimental data specific for individual proteins (mutational landscapes) and those for which data mixes different proteins from different organisms. Prediction task: as described in Methods; SubCellLoc: sub-cellular location, SecStr: secondary structure prediction (in 3 states: helix, strand, other). A number of sequences (i.e., proteins, note this is NOT the number of samples, e.g., for secondary structure prediction N proteins – number given in table - correspond to over 200*N residues in the data set): typical cross-validation using Train to optimize fine-tuning, Validation to optimize hyperparameters, and Test only to assess performance (Methods).

- The prediction task level per-residue and per-protein
- Sequence diversity data sets with many sequence-diverse proteins and data sets from mutational studies

DATABASES-2

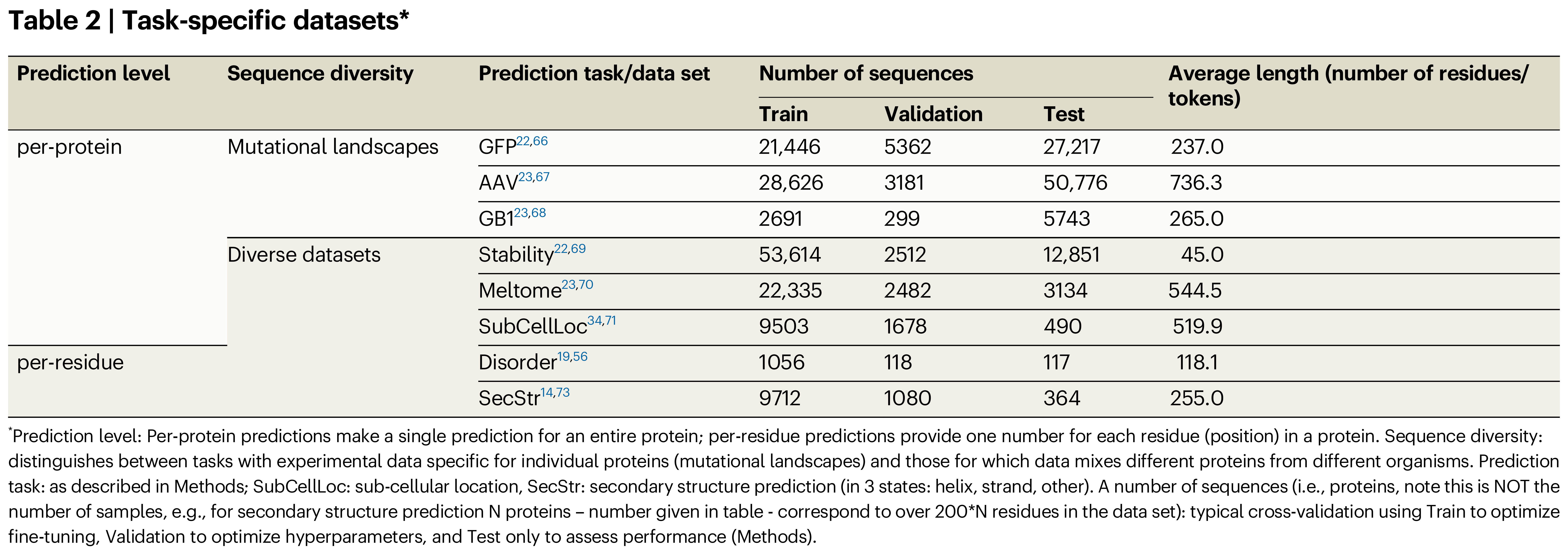


Table 2 | Task-specific datasets*

Prediction level	Sequence diversity	Prediction task/data set	Number of sequences			Average length (number of residues/tokens)
			Train	Validation	Test	
per-protein	Mutational landscapes	GFP[22,66]	21,446	5362	27,217	237.0
		AAV[23,67]	28,626	3181	50,776	736.3
		GB1[23,68]	2691	299	5743	265.0
	Diverse datasets	Stability[22,69]	53,614	2512	12,851	45.0
		Meltome[23,70]	22,335	2482	3134	544.5
		SubCellLoc[34,71]	9503	1678	490	519.9
per-residue		Disorder[19,56]	1056	118	117	118.1
		SecStr[14,73]	9712	1080	364	255.0

*Prediction level: Per-protein predictions make a single prediction for an entire protein; per-residue predictions provide one number for each residue (position) in a protein. Sequence diversity: distinguishes between tasks with experimental data specific for individual proteins (mutational landscapes) and those for which data mixes different proteins from different organisms. Prediction task: as described in Methods; SubCellLoc: sub-cellular location, SecStr: secondary structure prediction (in 3 states: helix, strand, other). A number of sequences (i.e., proteins, note this is NOT the number of samples, e.g., for secondary structure prediction N proteins – number given in table - correspond to over 200*N residues in the data set): typical cross-validation using Train to optimize fine-tuning, Validation to optimize hyperparameters, and Test only to assess performance (Methods).

- For mutational landscapes: regression tasks
  - For GFP (the green fluorescent protein) : fluorescence intensity
  - For AAV (adeno-associated virus) : the viability for packaging DNA payloads by mutating 28-amino acid window
  - For GB1 (the first binding domain of the protein G) : stability and binding affinity

DATABASES-3

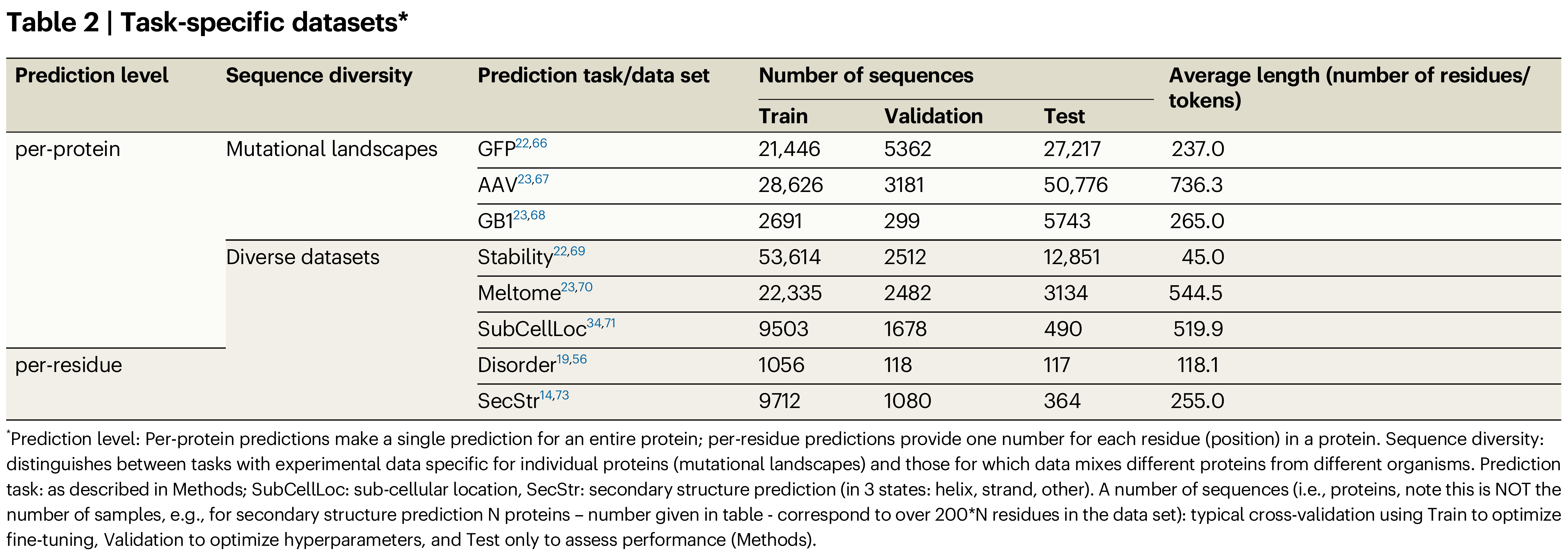


Table 2 | Task-specific datasets*

Prediction level	Sequence diversity	Prediction task/data set	Number of sequences			Average length (number of residues/ tokens)
			Train	Validation	Test	
per-protein	Mutational landscapes	GFP[22,66]	21,446	5362	27,217	237.0
		AAV[23,67]	28,626	3181	50,776	736.3
		GB1[23,68]	2691	299	5743	265.0
	Diverse datasets	Stability[22,69]	53,614	2512	12,851	45.0
		Meltome[23,70]	22,335	2482	3134	544.5
		SubCellLoc[34,71]	9503	1678	490	519.9
per-residue		Disorder[19,56]	1056	118	117	118.1
		SecStr[14,73]	9712	1080	364	255.0

*Prediction level: Per-protein predictions make a single prediction for an entire protein; per-residue predictions provide one number for each residue (position) in a protein. Sequence diversity: distinguishes between tasks with experimental data specific for individual proteins (mutational landscapes) and those for which data mixes different proteins from different organisms. Prediction task: as described in Methods; SubCellLoc: sub-cellular location, SecStr: secondary structure prediction (in 3 states: helix, strand, other). A number of sequences (i.e., proteins, note this is NOT the number of samples, e.g., for secondary structure prediction N proteins – number given in table - correspond to over 200*N residues in the data set): typical cross-validation using Train to optimize fine-tuning, Validation to optimize hyperparameters, and Test only to assess performance (Methods).

- For per-protein prediction, diverse datasets:
  - For Stability: measurements of protease susceptibility to digestion by de novo-designed mini-proteins
  - For Meltome: measuring thermostability for proteins from 13 species
  - For SubCellLoc: predict the sub-cellular location in one of ten classes

DATABASES-4

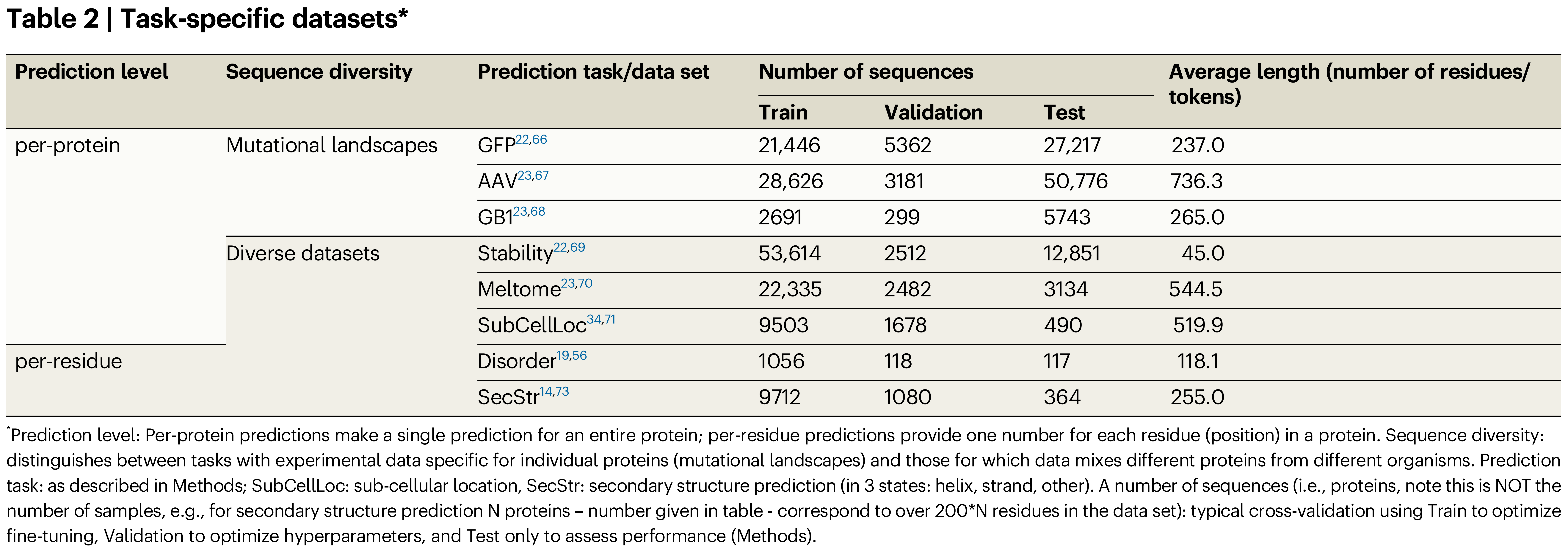


Table 2 | Task-specific datasets*

Prediction level	**Sequence diversity**	**Prediction task/data set**	**Number of sequences**			**Average length (number of residues/ tokens)**
			Train	**Validation**	**Test**	
per-protein	Mutational landscapes	GFP[22,66]	21,446	5362	27,217	237.0
		AAV[23,67]	28,626	3181	50,776	736.3
		GB1[23,68]	2691	299	5743	265.0
	Diverse datasets	Stability[22,69]	53,614	2512	12,851	45.0
		Meltome[23,70]	22,335	2482	3134	544.5
		SubCellLoc[34,71]	9503	1678	490	519.9
per-residue		Disorder[19,56]	1056	118	117	118.1
		SecStr[14,73]	9712	1080	364	255.0

*Prediction level: Per-protein predictions make a single prediction for an entire protein; per-residue predictions provide one number for each residue (position) in a protein. Sequence diversity: distinguishes between tasks with experimental data specific for individual proteins (mutational landscapes) and those for which data mixes different proteins from different organisms. Prediction task: as described in Methods; SubCellLoc: sub-cellular location, SecStr: secondary structure prediction (in 3 states: helix, strand, other). A number of sequences (i.e., proteins, note this is NOT the number of samples, e.g., for secondary structure prediction N proteins – number given in table - correspond to over 200*N residues in the data set): typical cross-validation using Train to optimize fine-tuning, Validation to optimize hyperparameters, and Test only to assess performance (Methods).

- For per-residue prediction:
  - For Disorder: predict CheZOD scores which quantify the level of intrinsic disorder for each residue in a protein
  - For Secondary Structure:

METHOD-1

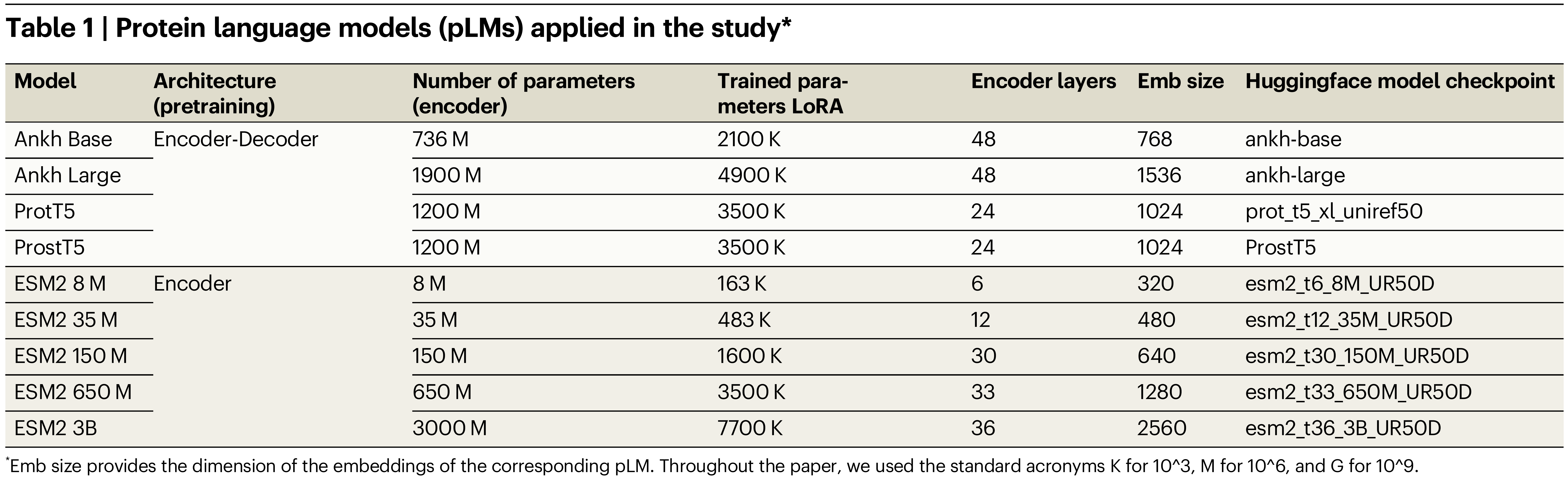


Table 1 | Protein language models (pLMs) applied in the study*

Model	Architecture (pretraining)	Number of parameters (encoder)	Trained parameters LoRA	Encoder layers	Emb size	Huggingface model checkpoint
Ankh Base	Encoder-Decoder	736 M	2100 K	48	768	ankh-base
Ankh Large		1900 M	4900 K	48	1536	ankh-large
ProtT5		1200 M	3500 K	24	1024	prot_t5_xl_uniref50
ProstT5		1200 M	3500 K	24	1024	ProstT5
ESM2 8 M	Encoder	8 M	163 K	6	320	esm2_t6_8M_UR50D
ESM2 35 M		35 M	483 K	12	480	esm2_t12_35M_UR50D
ESM2 150 M		150 M	1600 K	30	640	esm2_t30_150M_UR50D
ESM2 650 M		650 M	3500 K	33	1280	esm2_t33_650M_UR50D
ESM2 3B		3000 M	7700 K	36	2560	esm2_t36_3B_UR50D

*Emb size provides the dimension of the embeddings of the corresponding pLM. Throughout the paper, we used the standard acronyms K for 10^3, M for 10^6, and G for 10^9.

- ✦ We trained 615 individual prediction methods
  - ✦ 295 for fine-tuning,
  - ✦ 320 using frozen embeddings

from pre-trained pLMs comprising eight models.

METHOD-2

- We compared the performances between fine-tuning and pre-training.

- We trained each model-task combination multiple times with different random seeds.
- All results are averages over those runs.
- Validation set selected the best parameters.

- Final outcome: Supervised fine-tuning numerically increased performance for almost all combinations.

METHOD-3

- For pre-trained results:
  - We generated embeddings for all data sets.
    - For per-protein tasks: average over the sequence length for each protein
    - For per-residue tasks: all residue embeddings and their labels
  - We trained a single fully connected layer of size 32
    - input: embeddings
    - output (one of below):
      - a single value (regression)
      - another output layer with one neuron for each possible output class followed by a softmax layer to get a probability distribution
  - We repeated each training five times with different random seeds.

METHOD-4

- For fine-tuning results:
  - We added the same fully connected layer (size 32) to the pLM encoder as a prediction head.
  - For ProtT5 and Ankh, we used average pooling of the last hidden states over the sequence length dimension during training on per-protein tasks
  - For ESM2, we connected the prediction head only to the very first token.
  - The training was repeated three times with different random seeds.

  - To increase training efficiency and save time, we applied LoRA to all models.
  - For the smaller ESM2 models (up to the 650M version), we did full model fine-tuning.

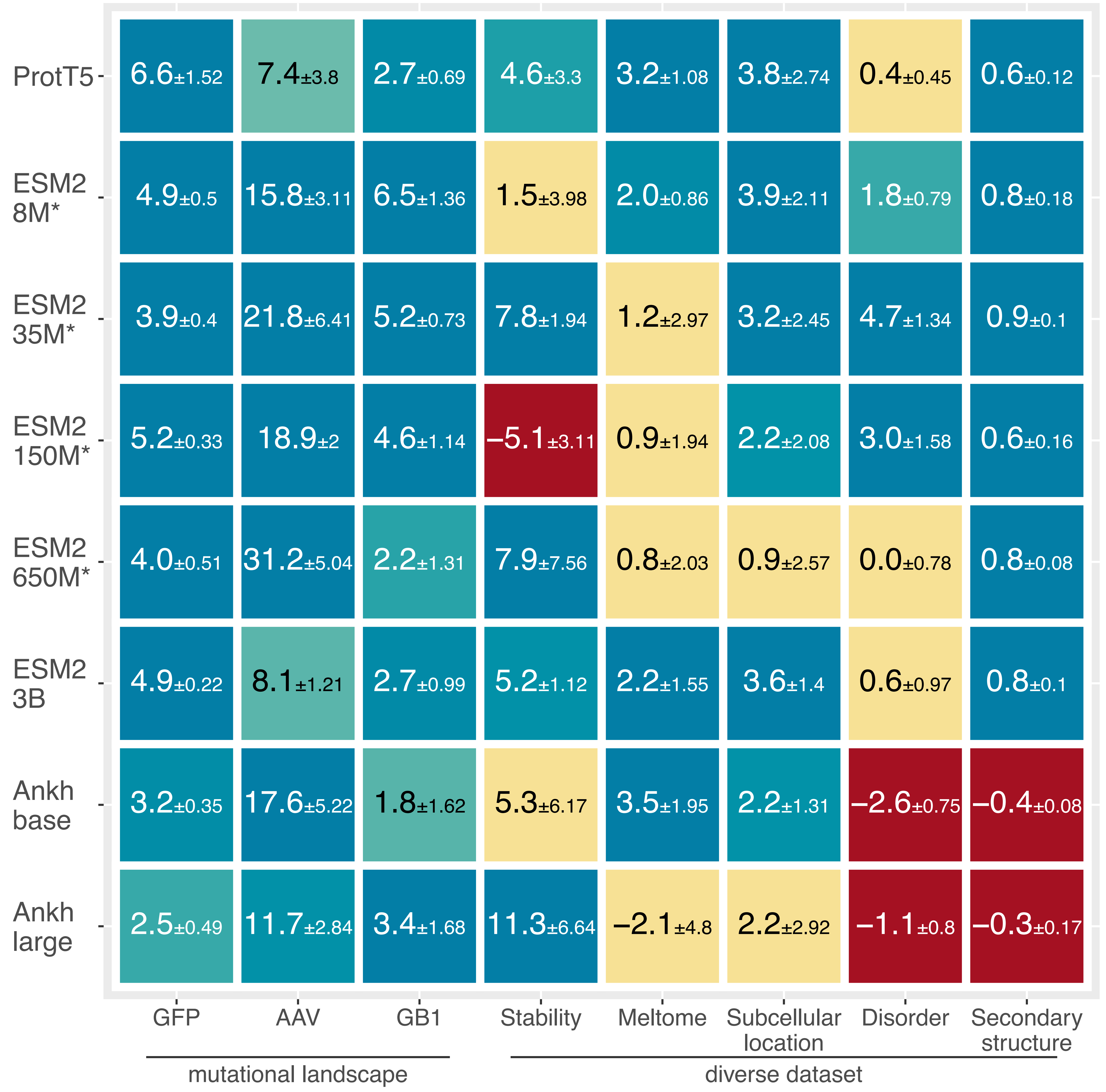
METHOD-5

- Presented results:

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

RESULTS-1

- Asterisks (*) mark fully fine-tuned models; the others were LoRA-optimized.
- Different performance measures are used for tasks:
    - the Spearman rank correlation (GFP, AAV, GB1, stability, meltome and disorder),
    - 10-class accuracy (Q10: sub-cellular location),
    - 3-class per-residue accuracy (Q3: secondary structure)
- Blue shows statistically significant increases. (~48/64)
- Yellow shows statistically insignificant changes. (11/64)
- Red shows supervised fine-tuning significantly decreased performance. (5/64)
- Green ?



	mutational landscape			diverse dataset				
	GFP	AAV	GB1	Stability	Meltome	Subcellular location	Disorder	Secondary structure
ProtT5	6.6±1.52	7.4±3.8	2.7±0.69	4.6±3.3	3.2±1.08	3.8±2.74	0.4±0.45	0.6±0.12
ESM2 8M*	4.9±0.5	15.8±3.11	6.5±1.36	1.5±3.98	2.0±0.86	3.9±2.11	1.8±0.79	0.8±0.18
ESM2 35M*	3.9±0.4	21.8±6.41	5.2±0.73	7.8±1.94	1.2±2.97	3.2±2.45	4.7±1.34	0.9±0.1
ESM2 150M*	5.2±0.33	18.9±2	4.6±1.14	−5.1±3.11	0.9±1.94	2.2±2.08	3.0±1.58	0.6±0.16
ESM2 650M*	4.0±0.51	31.2±5.04	2.2±1.31	7.9±7.56	0.8±2.03	0.9±2.57	0.0±0.78	0.8±0.08
ESM2 3B	4.9±0.22	8.1±1.21	2.7±0.99	5.2±1.12	2.2±1.55	3.6±1.4	0.6±0.97	0.8±0.1
Ankh base	3.2±0.35	17.6±5.22	1.8±1.62	5.3±6.17	3.5±1.95	2.2±1.31	−2.6±0.75	−0.4±0.08
Ankh large	2.5±0.49	11.7±2.84	3.4±1.68	11.3±6.64	−2.1±4.8	2.2±2.92	−1.1±0.8	−0.3±0.17

RESULTS-2 FINE-TUNING VS MODEL ARCHITECTURE

- To determine how much of the performance gains achieved by fine-tuning can be attributed to the model architecture itself and how much to the model's pre-training, we fine-tuned randomly initialized models.
- Due to the random starting point model training took much longer to converge.

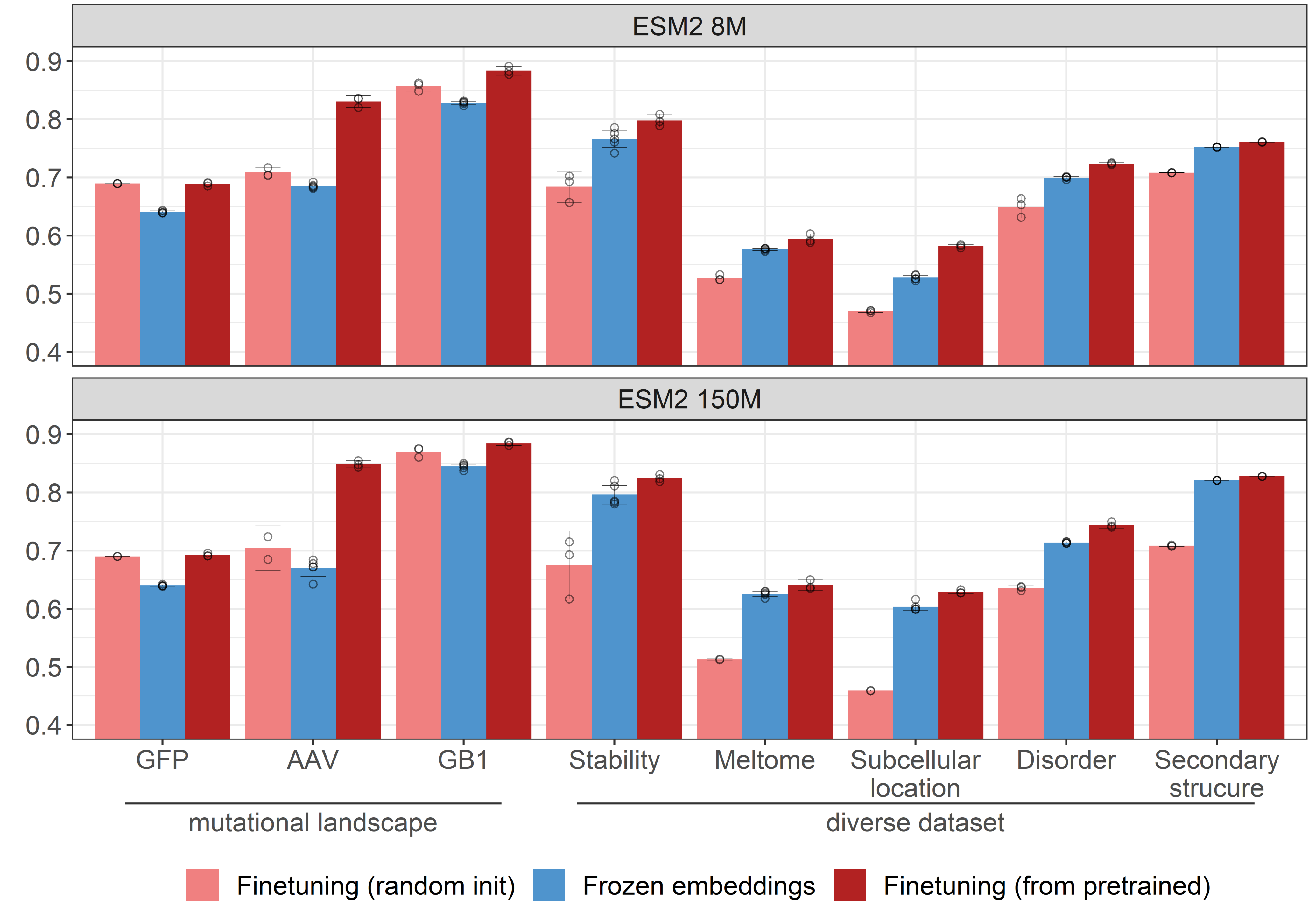


Figure S7. Model fine-tuning from random initialization. Values represent associated metrics with each dataset (Spearman ranking correlation for *GFP, AAV, GB1, Stability, Meltome* and *Disorder*; accuracy for 10-class, per-protein sub-cellular location and 3-class per-residue secondary structure), error bars mark the 95% confidence intervals (CI), calculated from multiple reruns of the same model training. Circles represent individual training runs. Source data are provided as a Source Data file.

RESULTS-3 FINE-TUNING VS MODEL ARCHITECTURE

- For diverse datasets, models fine-tuned from fully randomly initiated weights (light red - pink) do not reach the performance of predictors using pre-trained embeddings (blue).
- We concluded that for these datasets pre-training provides a major contribution.

- For the mutational landscapes, frozen embeddings performed worse than models fine-tuned from random initialization.
- For GFP, no difference between fine-tuning from pre-trained models and from random weights.
- For AAV and GB1, fine-tuning from pre-trained models preformed better than others.

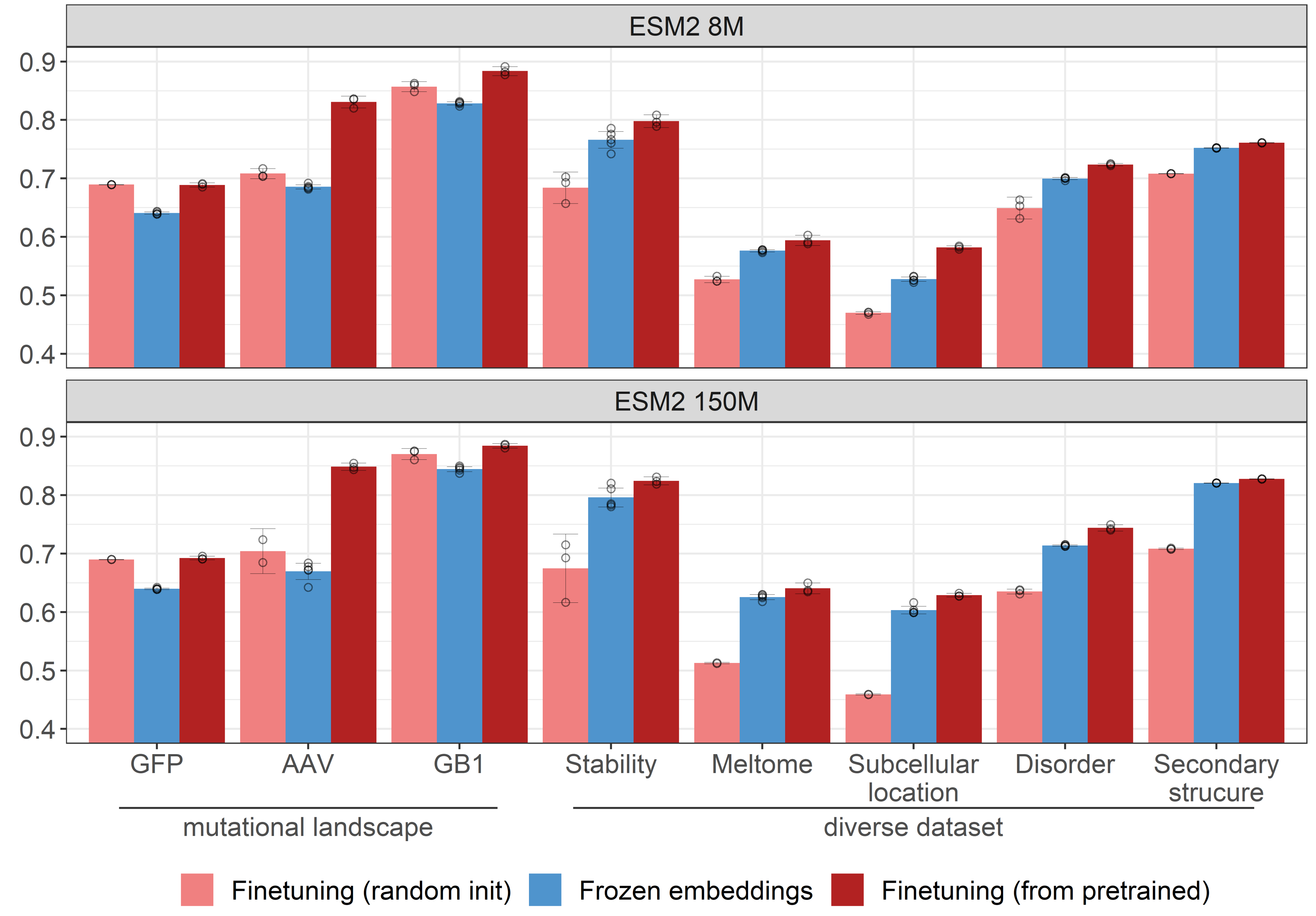
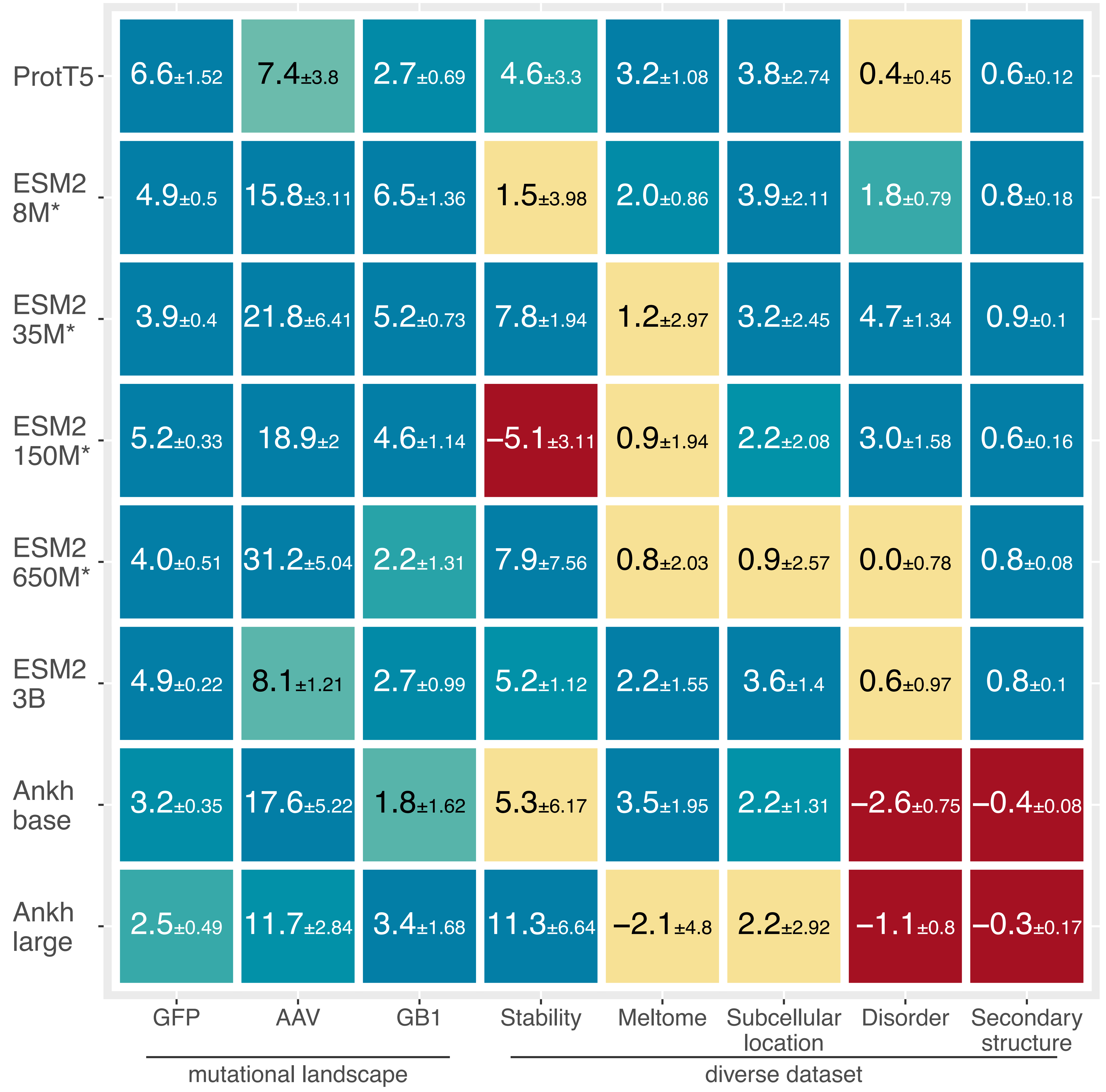


Figure S7. Model fine-tuning from random initialization. Values represent associated metrics with each dataset (Spearman ranking correlation for *GFP, AAV, GB1, Stability, Meltome* and *Disorder*; accuracy for 10-class, per-protein sub-cellular location and 3-class per-residue secondary structure), error bars mark the 95% confidence intervals (CI), calculated from multiple reruns of the same model training. Circles represent individual training runs. Source data are provided as a Source Data file.

RESULTS-4

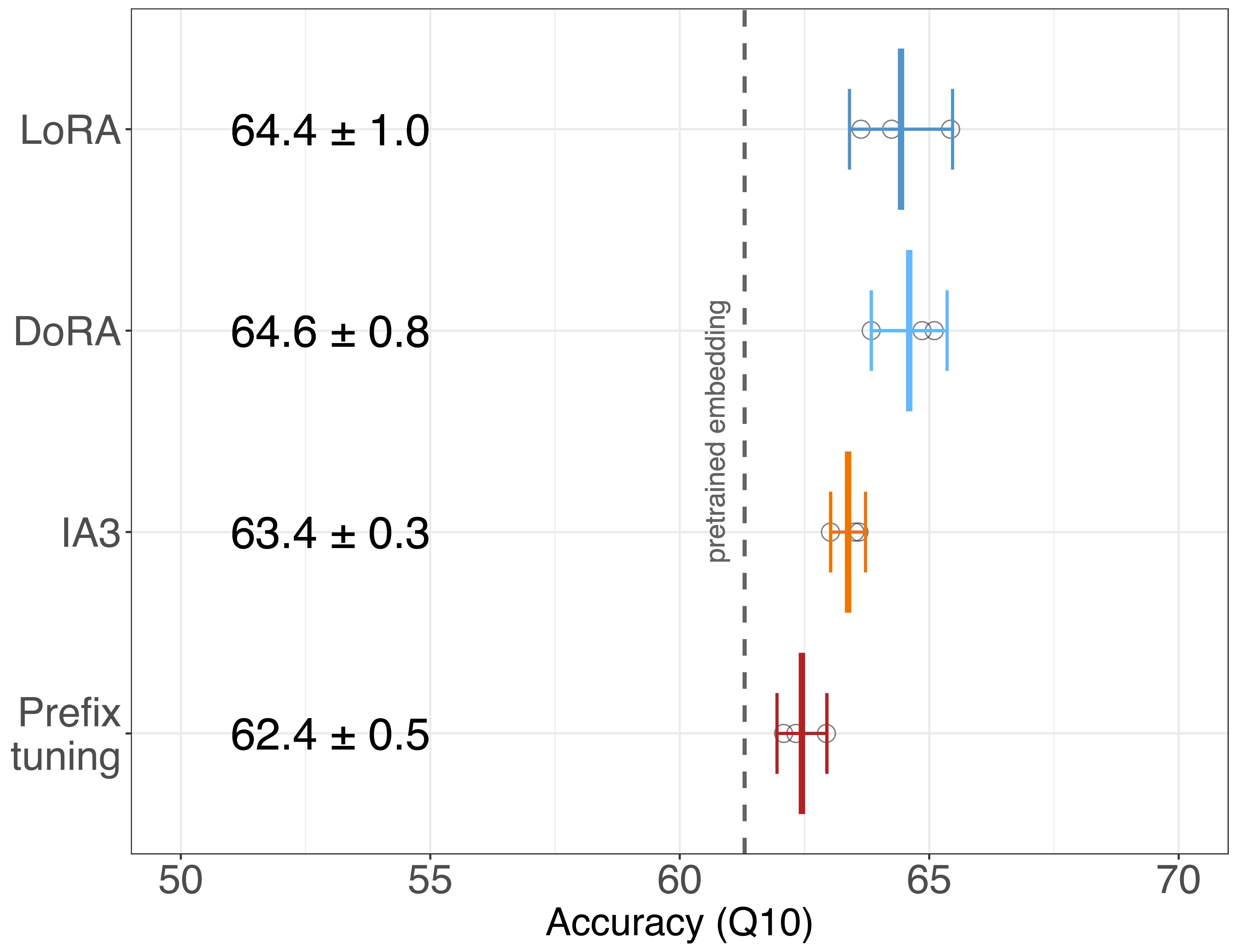
- For 5 of the 64 pLM/task combinations, fine-tuning performed worse (red).
- The observation of ESM2-150M on stability originated from instability in training picking a sub-optimal model.
- The other 4 originated from the Ankh pLM family on disorder and secondary structure. We were not able to track down a root cause here.



	mutational landscape			diverse dataset				
	GFP	AAV	GB1	Stability	Meltome	Subcellular location	Disorder	Secondary structure
ProtT5	6.6±1.52	7.4±3.8	2.7±0.69	4.6±3.3	3.2±1.08	3.8±2.74	0.4±0.45	0.6±0.12
ESM2 8M*	4.9±0.5	15.8±3.11	6.5±1.36	1.5±3.98	2.0±0.86	3.9±2.11	1.8±0.79	0.8±0.18
ESM2 35M*	3.9±0.4	21.8±6.41	5.2±0.73	7.8±1.94	1.2±2.97	3.2±2.45	4.7±1.34	0.9±0.1
ESM2 150M*	5.2±0.33	18.9±2	4.6±1.14	−5.1±3.11	0.9±1.94	2.2±2.08	3.0±1.58	0.6±0.16
ESM2 650M*	4.0±0.51	31.2±5.04	2.2±1.31	7.9±7.56	0.8±2.03	0.9±2.57	0.0±0.78	0.8±0.08
ESM2 3B	4.9±0.22	8.1±1.21	2.7±0.99	5.2±1.12	2.2±1.55	3.6±1.4	0.6±0.97	0.8±0.1
Ankh base	3.2±0.35	17.6±5.22	1.8±1.62	5.3±6.17	3.5±1.95	2.2±1.31	−2.6±0.75	−0.4±0.08
Ankh large	2.5±0.49	11.7±2.84	3.4±1.68	11.3±6.64	−2.1±4.8	2.2±2.92	−1.1±0.8	−0.3±0.17

RESULTS-5 DIFFERENT PEFT METHODS

- For ProtT5 and subcellular location prediction, we compared 3 parameter-efficient fine-tuning methods to LoRA.
- The fraction of trained model parameters were
  - 0.25% for LoRA
  - 0.28% for DoRA
  - 0.12% for IA3
  - 0.5% for Prefix tuning
- Despite these differences, runtimes for training and testing were within ±10% between methods.
- Overall, all fine-tuning methods improved, on average, over pre-trained embeddings.
- In terms of prediction performance, LoRA and DoRA outperformed IA3 and Prefix-tuning.
- As no method improved significantly over the well-established LoRA, we used it throughout our experiments.

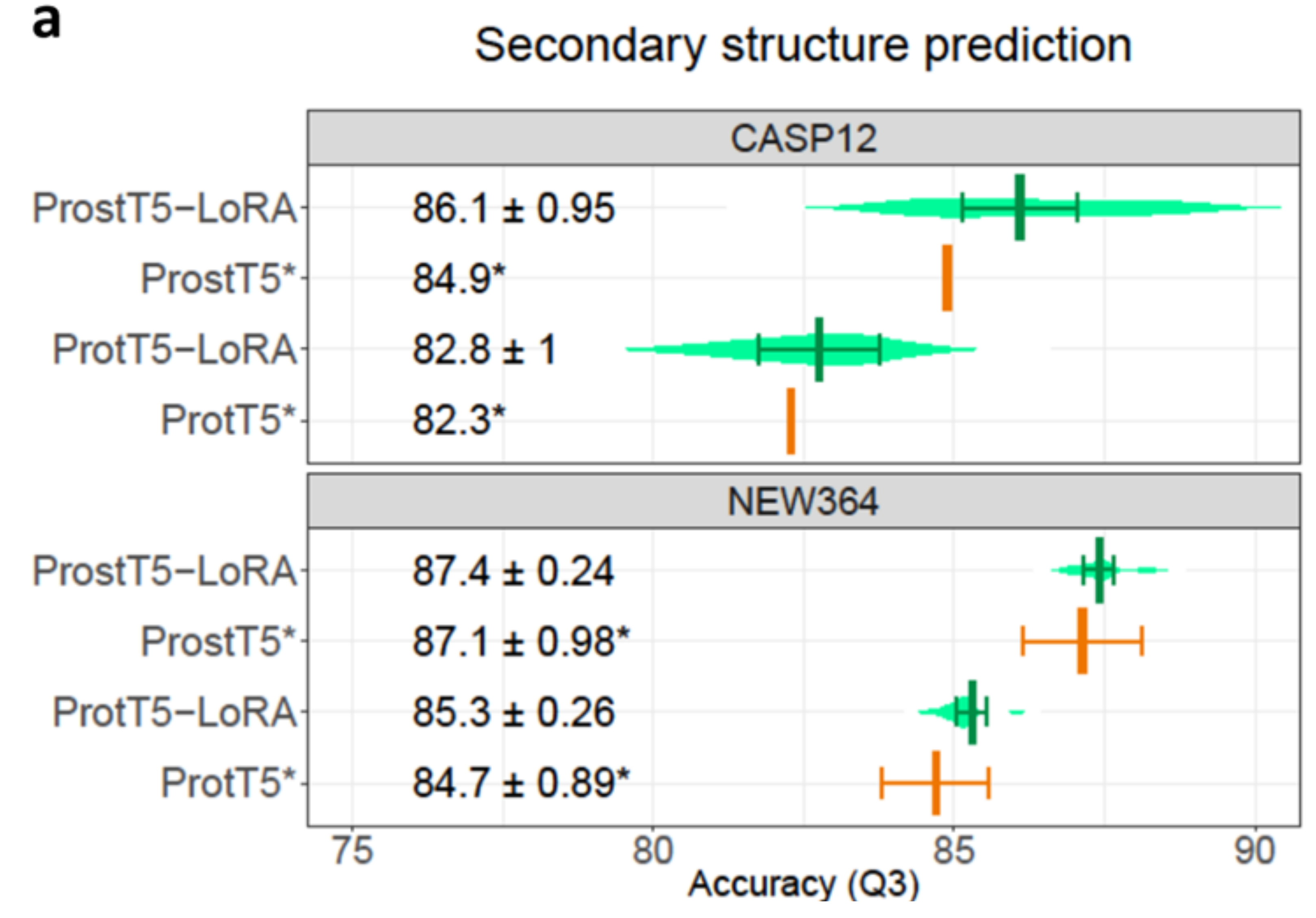


RESULTS-6 SECONDARY STRUCTURE PREDICTION

- For per-residue, three-class secondary structure prediction (helix, strand, other), fine-tuning improved only slightly.
- fine-tuning in green
- pre-trained embeddings in orange

- Two possible reasons:
  - secondary structure might already have been captured in unsupervised pre-training
  - performance may have reached an upper limit

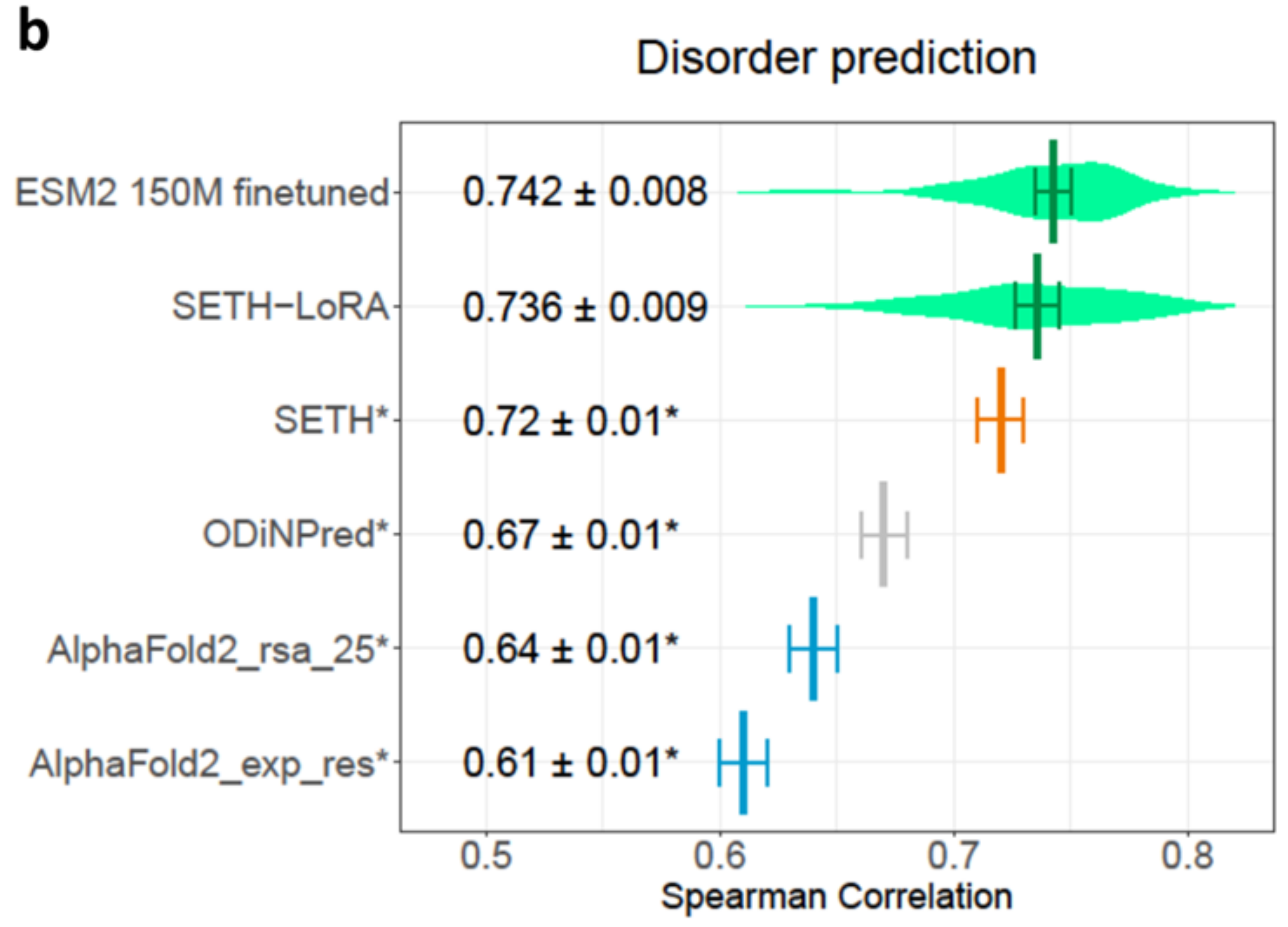
a



RESULTS-7 DISORDER PREDICTION

- ODiNPred: an MSA-based SOTA method
- SETH inputs ProtT5 embeddings into a two-layer CNN
- pLM-based SETH reached the level of ODiNPred in the prediction
- Keeping those hyper-parameters and adding LoRA fine-tuning (SETH-LoRA), improved performance (from Spearman 0.72 to 0.736).
- Fine-tuning the much smaller 150M parameter ESM2 model (Spearman: 0.742) improved overall solutions compared.

b



RESULTS-8 SUBCELLULAR LOCATION

- Most predictions of subcellular location input signals averaged over entire proteins.
- Embedding-based solutions do this through pooling.
- Light Attention (LA) improves over averaging by learning the optimal per-residue signal and combining this with the average.
- LoRA fine-tuning combined the advantage of a small model (fewer free parameters) with the learned, weighted averaging of LA.

- Therefore, LoRA fine-tuning numerically surpassed LA.
- But, the difference was statistically significant only at an 88% confidence interval.

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

- ProtT5 FFN with standard average pooling
- ProtT5 LA with light attention pooling
- ProtT5 FFN-LoRA with standard average pooling and LoRA

RESULTS-9 PREDICTING MUTATION LANDSCAPES

- For predicting mutation landscapes, fine-tuning any pLM succeeded.
- As differences between fine-tuned models were small, we averaged performance across all fine-tuned pLMs, and compared to homology-based inference (HBI, using MMseqs2 search) and to reference-free analysis (RFA).
- Blue shows the average performance across all fine-tuned pLMs
- Smaller and larger models performed alike on these tasks => use small, fine-tuned pLMs

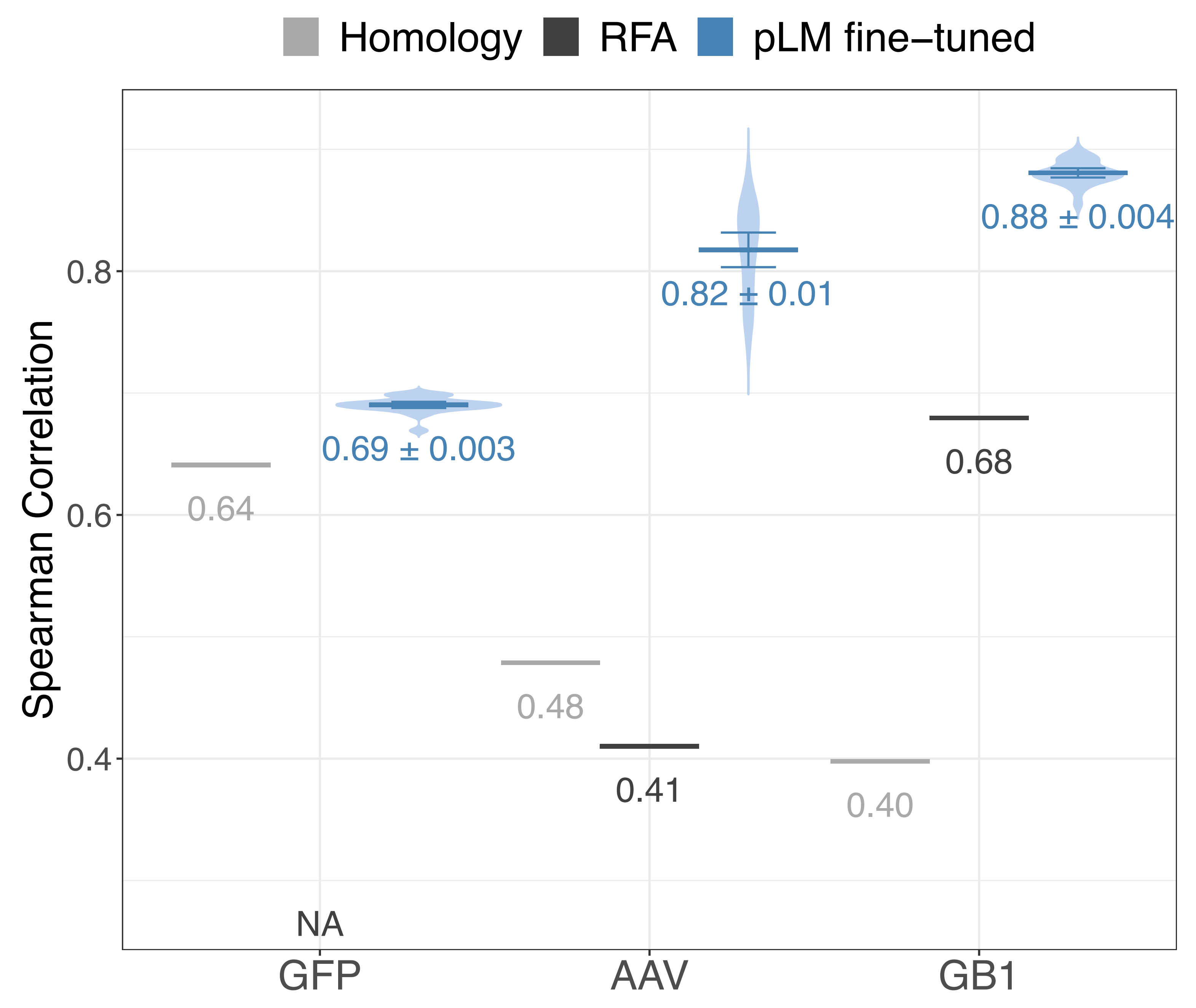


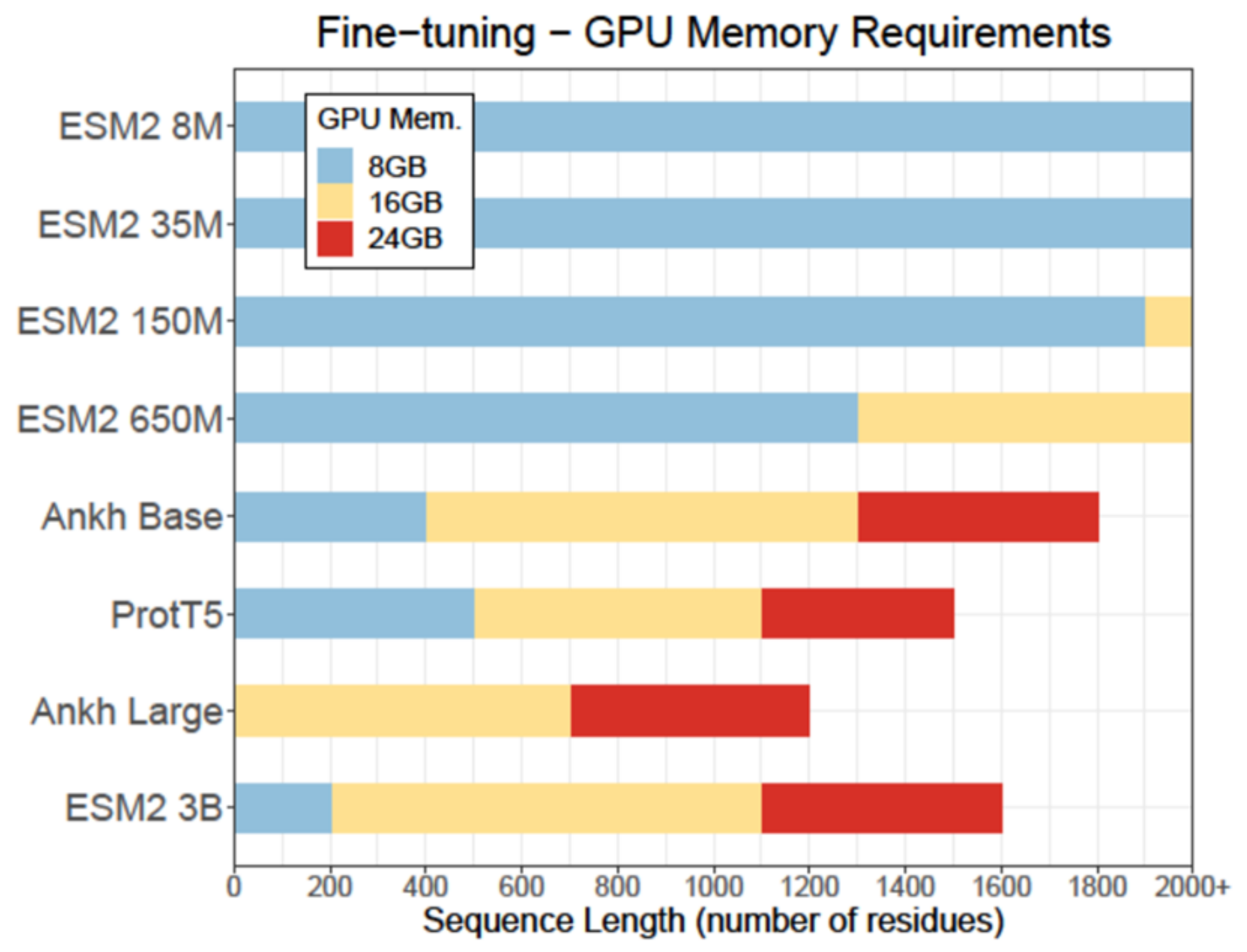
Fig. 4 | Simple methods limited for mutational effects. Blue: average performance across all fine-tuned pLMs (mean values with 95% CI, $n = 24$) with violin plots providing the underlying distribution; Gray: two simple baseline methods: Homology (HBI): MMseqs2[55] inferred the fitness value of test set proteins from most sequence-similar homolog in the training set. RFA (reference-free analysis[56]) fits models based on the assumption, that most of the mutational effects can be described as the sum of low-order effects. Source data are provided as a Source Data file.

RESULTS-10 LORA

- More parameters leads to more computational time and resources required.

- GPUs with beyond 40GB of memory will have sufficient memory to allow usage of all pLMs tested here.
- For less powerful hardware, mixed precision training nearly halved the required GPU memory without performance loss.
  - using half-precision floating point numbers
  - nearly halves memory requirements and speeds up calculations
- In addition, we applied gradient accumulation to reduce the actual on-device batch size as far as needed.
  - a technique to support larger batch sizes in case of limited GPU memory
- When even an on-device batch size of 1 was insufficient, we used DeepSpeed to offload the optimizer and potential parameters to CPU-reduced GPU memory requirements further. (CPU offloading)
  - the transfer of resource intensive computational tasks to a separate processor (CPU in this case)

b

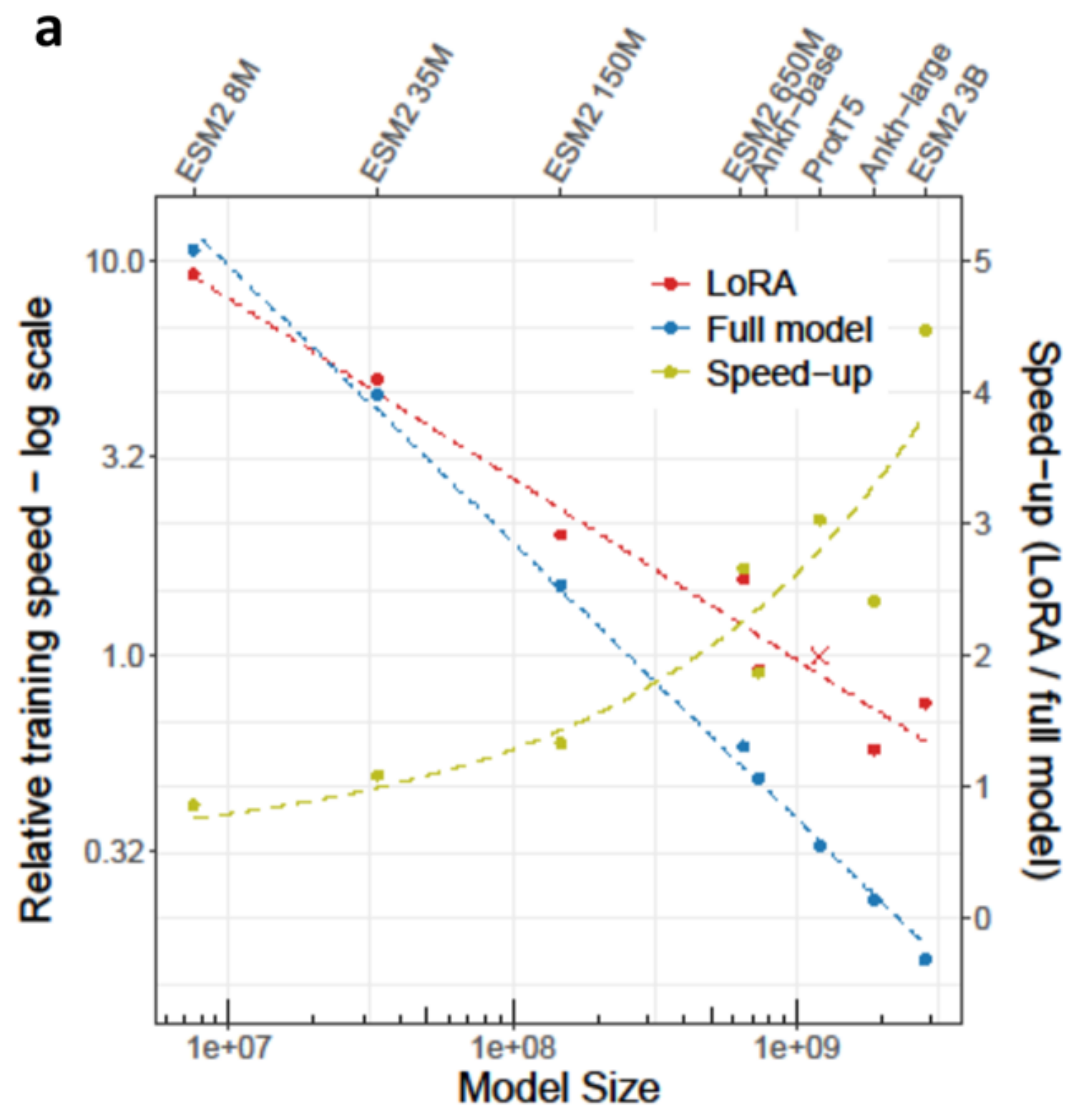


- maximum sequence length before the GPU runs out of memory (for 8, 18, and 24GB GPUs)

RESULTS-11 LORA

- ✦ As a trade-off, both gradient accumulation and CPU offloading slowed down training. Hence, both should be used cautiously.

- ✦ Both full model fine-tuning and parameter-efficient LoRA fine-tuning required the same amount of GPU memory and only differed in training speed when CPU offloading was utilized.



- ✦ Relative training speed of full fine-tuning (blue) and LoRA (red) is shown on a logarithmic scale, ProtT5 LoRA fine-tuning served as reference speed with value of 1 (x).

- ✦ The resulting speed-up for each model (olive) is shown on a normal scale.

FINE-TUNING RECIPE-1

- Before starting model training, prepare dataset splits
    - training (optimizing weights),
    - cross-training/validation (optimization of hyper-parameters, e.g., to decide between CNN and ANN),
    - and testing (only to estimate performance)
- Try to use all proteins
    - clustering by sequence identity using standard alignment methods such as MMseqs2
    - for structure-related tasks, using 3D clustering as realized by Foldseek
- To optimize the prediction of mutational landscapes for a single protein, it might be best to train on k-mers with k = 1 (single amino acid variants)

FINE-TUNING RECIPE-2

- To predict landscapes of mutational effects for specific proteins, we recommend
  - to first fine-tune a smaller pLM.
  - to optimize hyperparameters and head architectures on this smaller model.
- If done, explore additional improvements from larger pLMs.

- For the fine-tuning on diverse tasks, larger models mostly outperformed smaller models.
- Therefore, starting with raw embedding-based solutions to identify the best model and to then investigate different prediction heads appeared better than optimizing the fine-tuning directly.
- Applying parameter-efficient LoRA fine-tuning and optimizing hyperparameters for the selected model afterward.

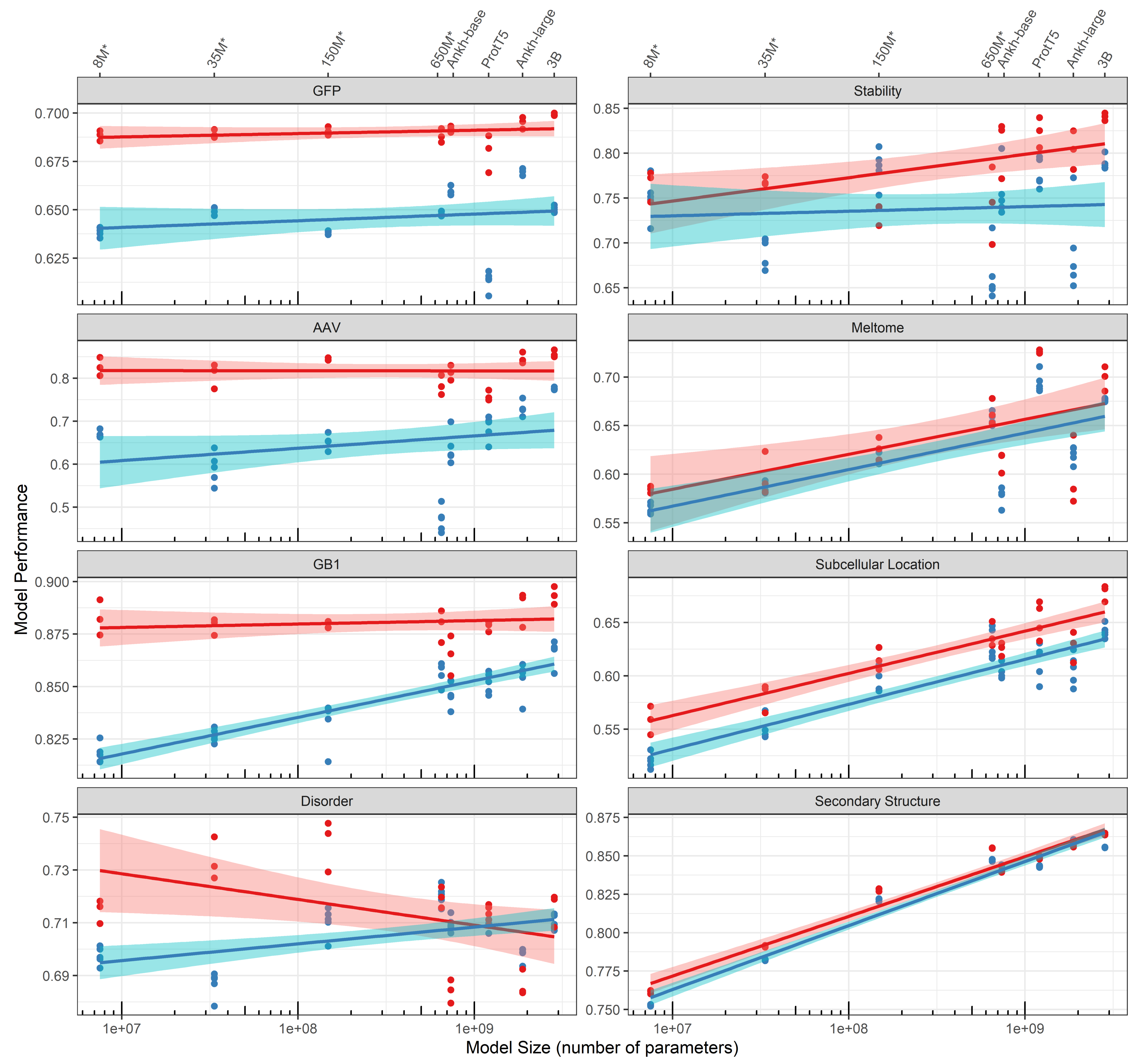


Figure S3. Model size impact per prediction task. Reuses the aggregated fine-tuning and pre-trained embedding results from Table S3 - S6, this time mapped to the model size with corresponding models shown at the top. Analog to Fig. 1 and Fig. S5 we report performance of fully fine-tuned models for smaller ESM2 models up to 650M (marked with *). Lighter colored areas are 95% confidence intervals. Source data are provided as a Source Data file.

FINE-TUNING RECIPE-3

- For our tasks, over-fitting mostly originated from data set characteristics.
- To mitigate over-fitting:
  - avoid imbalanced datasets,
  - providie sufficient high-quality training data,
  - choose smaller models for limited data sets.

- For computation: mixed precision training
- For memory: gradient accumulation and DeepSpeed’s CPU-offloading
- With all these measures in place, a single 16 GB GPU enables fine-tuning in many cases.

FINE-TUNING RECIPE-4

- Different PEFT methods:
    - Comparing different common PEFT methods did not suggest a clear winner.
    - LoRA was among the best solutions and was stable across our experiments.
    - The code-base provided by us simplifies experimenting with different PEFT approaches as it utilizes the Hugging Face PEFT framework.
    - We encourage you to compare different PEFT methods for your specific use cases.

- PEFT is memory efficient, but CPU-offloading could achieve the same.
- PEFT is also most compute-efficient for larger pLMs.

- We recommend LoRA fine-tuning for all models larger than ESM2 150 M.

CONCLUSION-1

- We applied fine-tuning to a diversity of prediction tasks clearly showing improvements, on average.
- The extent of this improvement varied by task and pLM model.
- The improvement was impacted by:
  - the amount of training data,
  - dataset balance,
  - model size, and
  - initial representation quality.

- Overall, our results revealed the gains initially observed in NLP from supervised task-specific fine-tuning of LLMs to also apply to large protein pLMs.

CONCLUSION-2

- The last hidden layer has been optimized for the unsupervised pre-training objective. This optimization might be suboptimal for any downstream task.
- PEFT (or finetuning in general) enables information from middle layers to flow to the last layer, making it accessible to downstream tasks.
- The transformer models might extract additional information directly from the task-specific training.
- FINAL WORD: We suggest to add supervised fine-tuning whenever applying transfer-learning.