Pretrained protein language model transfer learning: is the final layer representation what we want?

Francesca-Zhoufan Li
Department of Bioengineering
California Institute of Technology
Pasadena, CA 91125
fzl@caltech.edu

Ava P. Amini
Microsoft Research New England
Cambridge, MA 02142
avasoleimany@microsoft.com

Kevin K. Yang
Microsoft Research New England
Cambridge, MA 02142
yang.kevin@microsoft.com

Alex X. Lu
Microsoft Research New England
Cambridge, MA 02142
lualex@microsoft.com

Abstract

Large pretrained protein language models have improved protein sequence-to-function prediction. This often takes the form of transfer learning, where final-layer representations from large pretrained models are extracted for downstream tasks. Although pretrained models have been empirically successful, there is little current understanding of how the features learned by pretraining relate to and are useful for downstream tasks. In this work, we investigate whether transferring a partial model, by using the output from a middle layer, is as effective as full model transfer, and if so, whether successful transfer depends on the downstream task and model properties. Across datasets and tasks, we evaluate partial model transfer of pretrained transformer and convolutional neural networks of varying sizes. We observe that pretrained representations outperform the one-hot baseline for most tasks. More importantly, we find that representations from middle layers can be as effective as those from later layers. To our knowledge, our work is the first to report the effectiveness of partial model transfer for protein property prediction. Our results point to a mismatch between the pretraining and downstream tasks, indicating a need for more relevant pretraining tasks so that representations from later layers can be better utilized for downstream tasks.

1 Introduction

Proteins serve critical biological functions, including cellular structure, metabolism, and signaling. Understanding and engineering proteins is of great interest because it has the potential to impact human health, sustainable manufacturing, agriculture, and beyond [Lutz and Iamurri, 2018]. A protein's function is largely determined by its amino acid sequence, but the rules dictating sequence-to-function relationships are complex and poorly understood [Hedstrom, 2002, Alexander et al., 2009]. Because experimental approaches to determine protein functions from their sequences are time- and resource-intensive, the ability to computationally predict a protein's function from its amino acid sequence would enable more efficient protein engineering.

Recently, large pretrained protein language models have advanced our ability to predict protein function from sequence, especially when labeled training data is sparse [Rives et al., 2020, Yang et al., 2022, Elnaggar et al., 2022, Brandes et al., 2022, Alley et al., 2019, Rao et al., 2019]. These models pair large-scale protein sequence databases with unsupervised pretraining; the pretraining

Machine Learning for Structural Biology Workshop, NeurIPS 2022.

task is usually to reconstruct corrupted protein sequences [Bepler and Berger, 2021, Ofer et al., 2021]. Representations from pretrained protein language models outperform the one-hot baseline across protein engineering tasks [Yang et al., 2022, Wittmann et al., 2021, Dallago et al., 2021].

Although pretrained models have been empirically successful, there is little current understanding of how the features learned in pretraining are useful for downstream tasks. In the absence of this knowledge, the common practice is to transfer the entire model, using the output from the last layer as a representation for protein sequences of interest. We hypothesize that this might not be the best practice for two reasons.

First, previous work on transfer learning for images has shown that the middle layers of a model often transfer better to downstream tasks than the last layer, because earlier layers extract more general information, while later layers become more specific to the pretraining task [Gidaris et al., 2018, Jenni and Favaro, 2018]. This intuition may be applicable to protein sequence representations: the earlier layers may capture general information sufficient for the downstream task while the more specific features from later layers may not be effective for a specific downstream task. Second, we suspect that not all downstream tasks are related to the pretraining task in the same way. We expect downstream tasks that are more related to the pretraining task will benefit from the more specific features learned from later layers, but it is unclear how to quantify relatedness between downstream and pretraining tasks.

In this work, we investigate whether transferring a partial model, by using the output from a middle layer, is as effective as full model transfer, and if so, whether successful transfer depends on the downstream task and model properties. Across datasets and tasks, we evaluate partial model transfer of pretrained transformer and convolutional neural networks of varying sizes. We observe that pretrained representations outperform the one-hot baseline for most tasks, indicating that transfer learning is generally effective. More importantly, we find that representations from middle layers can be as effective as those from later layers, suggesting that downstream tasks may only be benefiting from shallow features learned earlier in a model's hierarchical representation. To our knowledge, our work is the first to systematically assess the effectiveness of partial architecture transfer on proteins, indicating a need for more relevant pretraining tasks so that representations from later layers can be fully utilized for downstream tasks.

2 Methods

2.1 Downstream tasks

To assess how transfer learning performance depends on downstream tasks, we evaluate performance over a diverse set of tasks, reflecting both local and global variation. In total, we test 7 protein function tasks from 4 datasets from the recent FLIP benchmark [Dallago et al., 2021], summarized in Table 1 and visualized in Figures A1 and A2.

The GB1 and AAV datasets measure the effects of local variation. GB1 is the 56 amino-acid B1 domain of protein G, an immunoglobulin-binding protein. The GB1 dataset covers binding measurements for simultaneous mutations of up to 4 epistatic sites [Wu et al., 2016]. VP1 is an adeno-associated virus (AAV) capsid protein, over 700 amino acids long [Bryant et al., 2021]. The AAV dataset measures the effects of sparsely sampled mutations across a contiguous 28 amino-acid region over the binding interface, including insertions and deletions, on viral viability.

The other two datasets measure global protein properties for sequences spanning different functional families and domains of life. The thermostability dataset measures the melting temperature of 48,000 proteins across 13 species ranging from archaea to humans [Jarzab et al., 2020]. Subcellular localization is a classification dataset where the goal is to predict the cell compartment to which a eukaryotic protein localizes [Armenteros et al., 2017, Stärk et al., 2021].

To determine the impact of distribution shifts, we analyze different train and test splits of the two local protein engineering datasets, including sampled (in-distribution) and out-of-distribution splits, as shown in Table A2, Figure A1, and Figure A2. Out-of-distribution splits more closely resemble protein engineering applications where a few low-functioning variants with a limited number of mutations are initially generated, but high-functioning variants across the larger sequence space are the engineering end goal. For GB1, we test 3 different splits:

Table 1: Summary of downstream tasks

Dataset	Description	Variation	Task type
GB1	Immunoglobulin binding	Local	Regression
AAV	Viral viability	Local	Regression
Thermostability	Melting temperature	Global	Regression
Subcellular localization (scl)	Cellular location	Global	Classification

- Sampled: sequences randomly partitioned between 80% training and 20% testing
- Low vs high: an out-of-distribution task. Models are trained on mutants with function lower than the parent and tested on those with higher function.
- Two vs rest: an out-of-distribution task with fewer training samples. Models are trained on single and double mutants and tested on triple and quadruple mutants.

For AAV, we test 2 different splits:

- Two vs many: an out-of-distribution split. Models are trained on single and double mutants and tested on variants with three or more mutations.
- One vs many: an out-of-distribution task with fewer training samples. Models are trained on single mutants and tested on variants with more mutations.

2.2 Pretrained models

To evaluate the effectiveness of transferring a partial model, we systematically evaluate the performance of pretrained models on the aforementioned downstream tasks. We consider 2 different pretrained model architectures, transformer and convolutional models. For the transformer architecture, we use Evolutionary Scale Modeling (ESM) [Rives et al., 2020]. For the convolutional architecture, we use Convolutional Autoencoding Representations of Proteins (CARP) [Yang et al., 2022]. Both are pretrained with UniRef50 on masked language modeling, a pretraining task that requires models to reconstruct corrupted sequences. To test the effect of model size, we evaluate architectures with different numbers of layers and parameters for both architectures (details in Table A1). Due to the sequence length limit of the ESM-1b transformer model, the first and last 511 amino acids are taken for all sequences exceeding 1022 amino acids. This length restriction chiefly impacts the subcellular localization dataset: targeting signals often occur at the N- or C-terminal, and we reason that taking both terminals preserves biologically-relevant signals.

2.3 Layer-by-layer evaluation

To assess the performance of partial model transfer, we extract feature representations after every transformer or convolutional block of each model. We mean pool along the length dimension of the representations from each layer and train downstream linear models with these representations. Layer 0 corresponds to the input embedding before going through any model layers.

For the regression tasks, we train ridge regression models with Scikit-learn [Buitinck et al., 2013]. We tune the hyperparameter α based on the validation set. As protein engineers often seek to identify top-ranked mutants as opposed to predicting the absolute function of mutations, we use ranking as the primary metric for the validation and test datasets, but use mean-squared error (MSE) for the training set. For the classification task, we train linear classification models with mini-batches in PyTorch and perform early stopping based on the validation set. We evaluate the test set with accuracy (acc) and area under the receiver operating characteristic curve (ROC-AUC) metrics, in addition to cross-entropy loss.

2.4 Baselines and ablations

To evaluate if learned representations improve over the simplest representation, we compare pretrained representations to linear models trained on one-hot encodings. To disentangle the effects of pretraining and architecture, we randomly initialize model weights for the layer-by-layer evaluation (hereafter referred to as "random init"). Previous work in medical image transfer analysis [Matsoukas et al.,

2022] further proposed randomizing layers to match the distribution of pretrained weights. This helps further understand if features are being re-used from pretrained models, or if the features are simply sensibly initialized to interact well with the model architecture (e.g. magnitudes compatible with activation functions, sparsity, etc.). In our setting, we implement this by randomly permuting the learned weights within each layer (hereafter referred to as "stat transfer").

3 Results

3.1 ESM-1b performance saturates early for most tasks

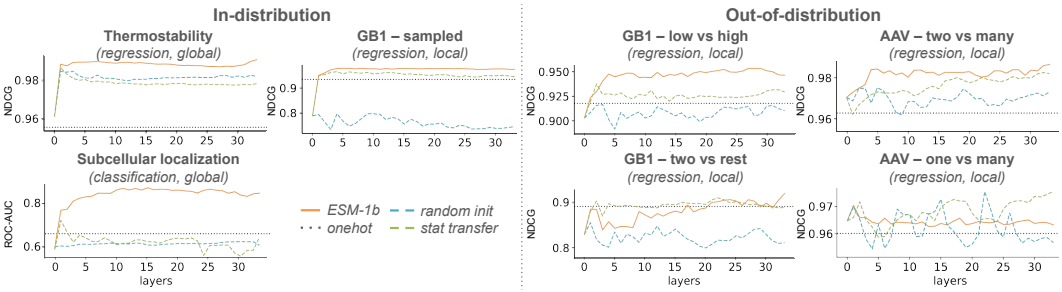


Figure 1: Performance of ESM-1b layer transfer on in-distribution and out-of-distribution tasks.

We first assess how transfer learning performance depends on the downstream task. For regression tasks, we use the Normalized Discounted Cumulative Gain (NDCG), which reflects ranking quality, as our performance metric; for the classification task, we use ROC-AUC. Figure 1 shows ESM-1b transfer learning performance on all 7 tasks. For most tasks, ESM-1b (solid orange lines) performance plateaus after the first five layers. However, ESM-1b outperforms the one-hot baseline (dotted black lines), suggesting that even though performance saturates early, the pretrained features are still informative for downstream tasks. In addition, ESM-1b with pretrained weights outperforms the ESM-1b architecture with random init (dashed blue lines). In fact random init does not always outperform the one-hot baseline. Thus, performance is not due to inductive biases of the transformer architecture alone, confirming that pretraining has benefits in capturing features relevant to protein function prediction. The pretrained weights usually outperform stat transfer (dashed green lines), most notably on the subcellular localization classification task. However, for the GB1 – sampled and AAV – two vs many tasks, stat transfer performs better than random init, and almost on par with pretraining. This suggests that for some tasks, such as subcellular localization, pretraining is required to generate linearly separable features. In contrast, for the GB1 – sampled task, a better weight initialization is sufficient to generate linearly separable features.

Next, we repeated our experiments with more challenging out-of-distribution train and test splits (GB1 – low vs high, GB1 – two vs rest, AAV – one vs many, AAV – two vs many). Figure 1 shows that for the GB1 and AAV datasets, the out-of-distribution tasks also exhibit a similar early peaking behavior to the in-distribution tasks, except that most representations perform poorly on the most challenging AAV – one vs many task. We hypothesize that the AAV – one vs many task does not provide enough data to fit a downstream model.

3.2 Early saturation holds across different ESM-1 model sizes

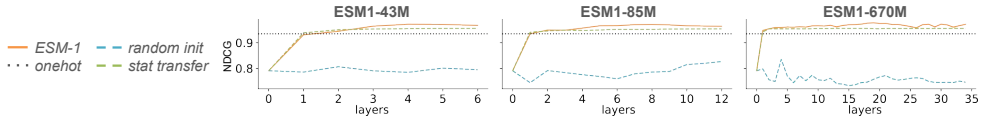


Figure 2: On the GB1 – sampled dataset, ESM-1 partial transfer performance saturates at early layers.

We observe that early saturation of performance occurs across pretrained ESM-1 models, regardless of parameter size and number of layers, and for both regression and classification tasks (Figure 2, Section A.3). For in-distribution tasks such as GB1 – sampled, we observe that the highest NDCG

achieved across model sizes is similar, suggesting that for these tasks larger models do not learn more informative features than smaller models.

3.3 Convolutional models also saturate early, but show stronger inductive biases

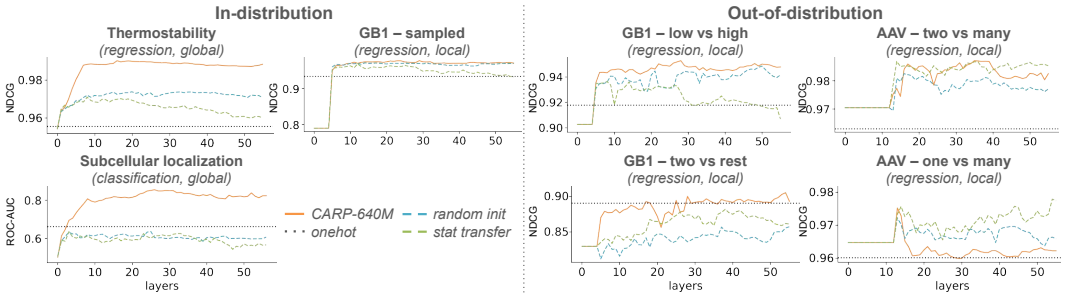


Figure 3: Performance of CARP-640M layer transfer on in-distribution and out-of-distribution tasks.

Next, we interrogate how architecture interacts with layer-by-layer performance by repeating the analysis for CARP-640M, as shown in Figure 3. For many tasks, we observe a more abrupt increase in performance, increasing from performance worse than one-hot to optimal performance rapidly. We hypothesize this is due to the inductive bias of convolutional neural networks: at some point, the receptive field becomes sufficiently large to resolve the information necessary for the downstream tasks. Consistent with this hypothesis, we primarily observe this behavior for datasets where all sequences are identical in length (GB1 and AAV), while datasets with a variety of lengths (thermostability and subcellular localization) exhibit more gradual increases in performance. We observe that for some downstream tasks, performance degrades as layer depth increases. This observation is in contrast to the transformer architectures, where performance past the saturation point is more stable. This trend is especially significant for the AAV – one vs many task in Figure 3. We also observe more subtle trends with the NDCG metric on the thermostability and subcellular localization datasets, and this trend is made more evident by considering the Spearman's rank correlation for the thermostability dataset (Figure A10). Overall, the decreasing generalization from the pretraining task to downstream task suggests that convolutional architectures may be more prone to overadapting to the pretraining task. Additionally, due to the inductive bias of convolutional neural networks towards motif-finding, stat transfer diminishes CARP-640M's performance more significantly than ESM-1b's, as seen most clearly in the thermostability task (Figures 1, 3). Conversely, the noticeable improvement of stat transfer over random init for GB1 – sampled with ESM-1b is no longer observed with CARP-640M (Figure 1, 3). Similarly to ESM, we observe that early saturation occurs across pretrained CARP model sizes and for both regression and classification tasks, as shown in Figure 4 and Section A.3.

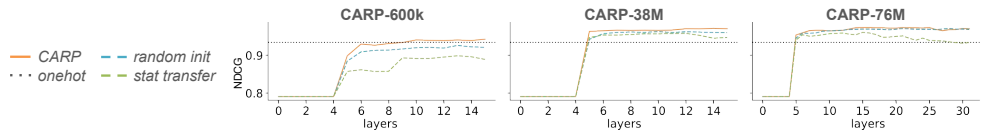


Figure 4: On the GB1 – sampled task, CARP partial transfer performance saturates at early layers.

4 Discussion

In this work, we systematically evaluate partial model transfer from large pretrained protein language models to various downstream protein function prediction tasks. We observe that pretrained models outperform their random init and stat transfer counterparts as well as the one-hot baseline for most tasks, indicating successful transfer learning from the sequence reconstruction pretraining task to various downstream protein function prediction tasks. However, we observe consistent early peaking behavior for both ESM and CARP, suggesting that most generalizable features are learned in early layers and that features from later layers are not always optimal for downstream tasks. Critically, this work points to a mismatch between language model pretraining and downstream protein function

prediction tasks. For many tasks only low-level features generated by early layers contribute to performance on downstream tasks, indicating that more relevant pretraining tasks are needed to fully utilize the capacity of the pretrained model on downstream tasks.

References

Stefan Lutz and Samantha M Iamurri. Protein engineering: Past, present, and future. *Methods in Molecular Biology*, 2018. doi: 10.1007/978-1-4939-7366-8_1.

Lizbeth Hedstrom. Serine protease mechanism and specificity. *Chem. Rev.*, 2002. doi: 10.1021/cr000033x.

Patrick A. Alexander, Yanan He, Yihong Chen, John Orban, and Philip N. Bryan. A minimal sequence code for switching protein structure and function. *PNAS*, 2009. doi: 10.1073/pnas.0906408106.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv preprint*, 2020. doi: https://doi.org/10.1101/622803.

Kevin K. Yang, Alex X. Lu, and Nicolo Fusi. Convolutions are competitive with transformers for protein sequence pretraining. *MLDD workshop, ICLR 2022*, 2022.

Ahmed Elnaggar, Michael Heinzinge, Christian Dallago, and et al. ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

Nadav Brandes, Dan Ofer, Yam Peleg, Nadav Rappoport, and Michal Linial. ProteinBERT: a universal deep-learning model of protein sequence and function. *Bioinformatics*, 38(8):2102–2110, 2022. doi: 10.1093/bioinformatics/btac020.

Ethan C. Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M. Church. Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 2019. doi: 10.1038/s41592-019-0598-1.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Xi Chen, John Canny, Pieter Abbeel, and Yun S Song. Evaluating protein transfer learning with TAPE. *bioRxiv*, 2019. doi: doi:https://doi.org/10.1101/676825.

Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell Systems*, 2021. doi: 10.1016/j.cels.2021.05.017.

Dan Ofer, Nadav Brandes, and Michal Linial. The language of proteins: NLP, machine learning & protein sequences. *Computational and Structural Biotechnology Journal*, 2021. doi: 10.1016/j.csbj.2021.03.022.

Bruce J. Wittmann, Yisong Yue, and Frances H. Arnold. Informed training set design enables efficient machine learning-assisted directed protein evolution. *Cell Systems*, 2021. doi: 10.1016/j.cels.2021.07.008.

Christian Dallago, Jody Mou, and et al. FLIP: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, 2021. doi: 10.1101/2021.11.09.467890.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv*, 2018. doi: arXiv:1803.07728v1.

Simon Jenni and Paolo Favaro. Self-supervised feature learning by learning to spot artifacts. *arXiv*, 2018. doi: arXiv:1806.05024v1.

Nicholas C Wu, Lei Dai, C Anders Olson, James O Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *eLife*, 2016. doi: 10.7554/eLife.16965.

Drew H Bryant, Ali Bashir, Sam Sinai, Nina K Jain, Pierce J Ogden, Patrick F Riley, George M Church, Lucy J Colwell, and Eric D Kelsic. Deep diversification of an AAV capsid protein by machine learning. *Nature Biotechnology*, 2021. doi: https://doi.org/10.1038/s41587-020-00793-4.

Anna Jarzab, Nils Kurzawa, Thomas Hopf, Matthias Moerch, Jana Zecha, Niels Leijten, Yangyang Bian, Eva Musiol, Melanie Maschberger, Gabriele Stoehr, et al. Meltome atlas—thermal proteome stability across the tree of life. *Nature methods*, 17(5):495–503, 2020.

José Juan Almagro Armenteros, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, 33(21):3387—-3395, 2017. doi: 10.1093/bioinformatics/btx431.

Hannes Stärk, Christian Dallago, Michael Heinzinger, and Burkhard Rost. Light attention predicts protein location from the language of life. *Bioinformatics Advances*, 1(1):vbab035, 2021. doi: 10.1093/bioadv/vbab035.

Lars Buitinck, Gilles Loupp, and et al. API design for machine learning software: experiences from the scikit-learn project. *eLife*, 2013. doi: 10.48550/arXiv.1309.0238.

Christos Matsoukas, Johan Fredin Haslum, Moein Sorkhei, Magnus SÖderberg, and Kevin Smith. What makes transfer learning work for medical images: Feature reuse & other factors. *arXiv*, 2022. doi: arXiv:2203.01825v2.

A Appendix

A.1 Pretrained models

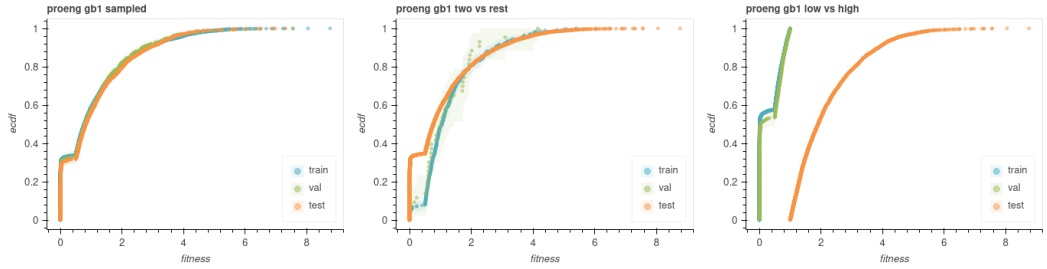
Table A1: pretrained model summary

Name	Architecture	# Layer	# Parameters	Embedding dimension
esm1_t6_43M_UR50S	ESM-1 transformer	6	43M	768
esm1_t12_85M_UR50S	ESM-1 transformer	12	85M	768
esm1_t34_670M_UR50S	ESM-1 transformer	34	670M	1280
esm1b_t33_650M_UR50S	ESM-1b transformer	33	650M	1280
carp_600k	CARP convolutional	16	600k	128
carp_38M	CARP convolutional	16	38M	1024
carp_76M	CARP convolutional	32	76M	1024
carp_640M	CARP convolutional	56	640M	1280

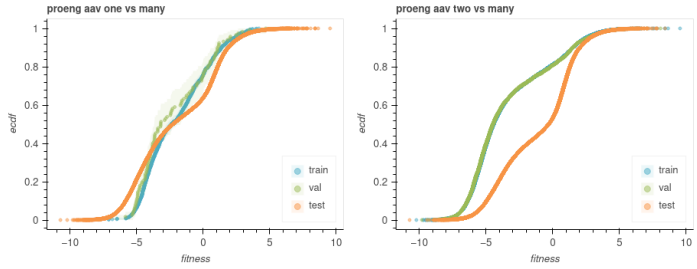
A.2 Dataset analysis

Table A2: Downstream task summary

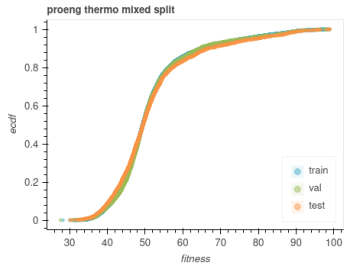
Task	Task type	Split type	# Train	# Val	# Test	Model type (# class)
GB1 – sampled	Local	In sample	6289	699	1745	Scikit-learn ridge regression
GB1 – low vs high	Local	Out of distribution	4580	509	3644	Scikit-learn ridge regression
GB1 – two vs rest	Local	Out of distribution (fewer training samples)	381	43	8309	Scikit-learn ridge regression
AAV – two vs many	Local	Out of distribution	28626	3181	50776	Scikit-learn ridge regression
AAV – one vs many	Local	Out of distribution (fewer training samples)	1053	117	81413	Scikit-learn ridge regression
Thermostability	Global	In sample	22335	2482	3134	Scikit-learn ridge regression
Subcellular localization (scl)	Global	In sample	9503	1678	385	PyTorch linear classifier (10)



(a) GB1 splits: sampled, two vs rest (out-of-distribution with fewer training samples), and low vs high (out of distribution)

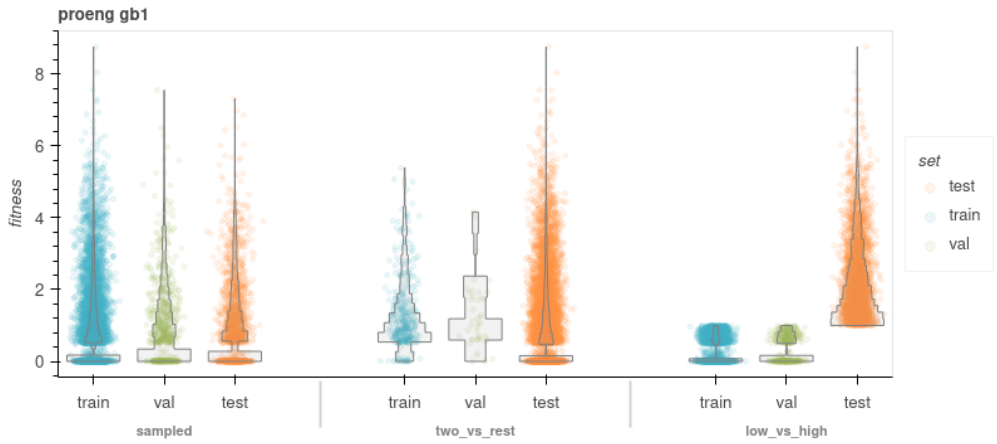


(b) AAV splits: one vs many (out-of-distribution with fewer training samples), two vs many (out of distribution)

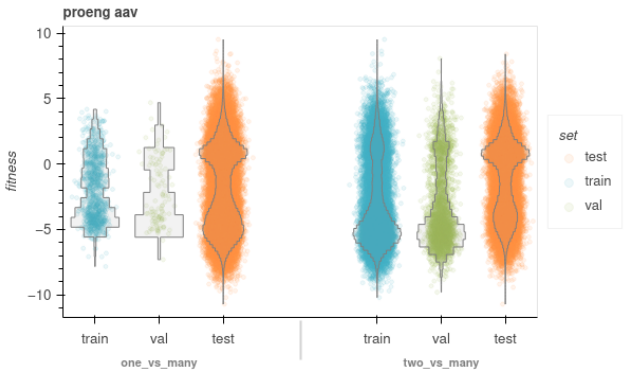


(c) Thermostability split: mixed

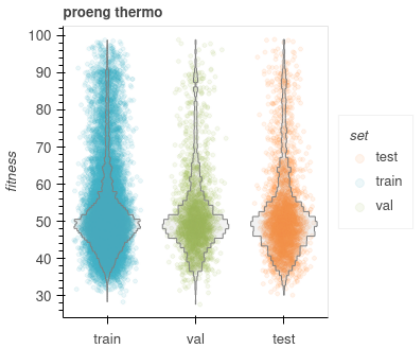
Figure A1: Empirical cumulative distribution functions for the 7 tasks.



(a) GB1



(b) AAV

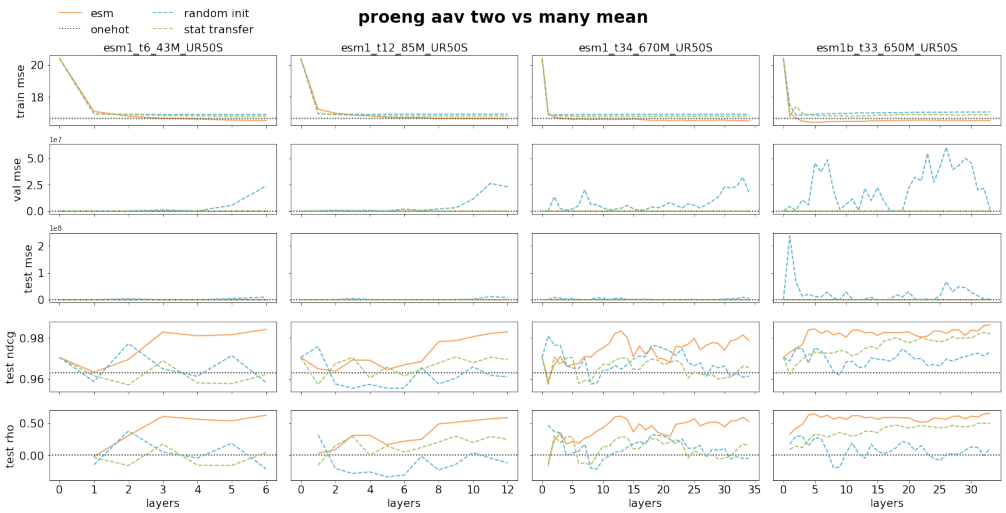


(c) Thermostability

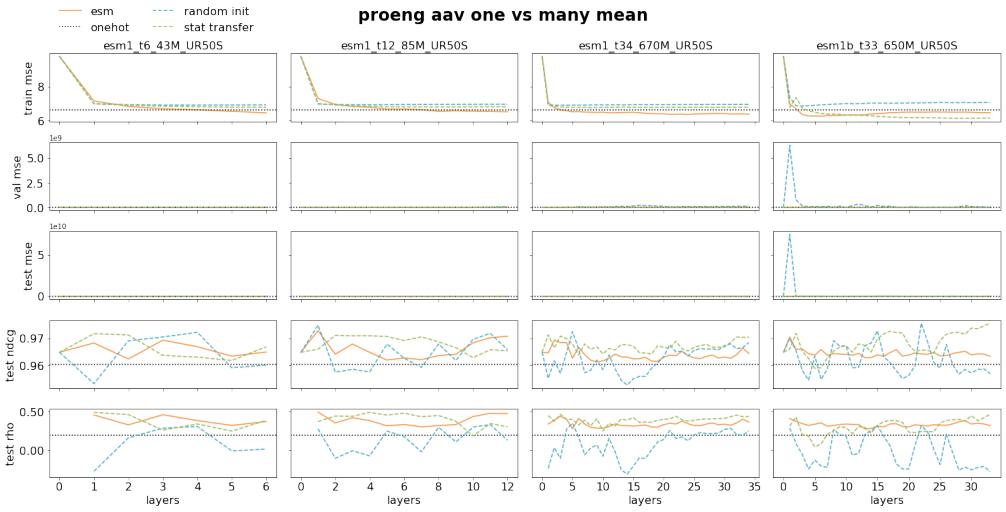
Figure A2: Strip-histogram for protein engineering datasets

A.3 Full results for ESM and CARP

All results have 5 rows and 4 columns. For ESM, the first three columns corresponds to pretraining with an ESM-1 model, ranging from the smallest architecture with the fewest number of layers and parameters to the largest. The last column corresponds to the ESM-1b model. For CARP, the columns corresponds to pretraining with a CARP model, ranging from the smallest architecture with the fewest number of layers and parameters to the largest. For regression tasks, (AAV, GB1, and thermostability), the rows are in the order of MSE for training, validation, and testing, followed by testing NDCG, and testing Spearman's rank correlation. For classification (subcellular localization), the rows are cross-entropy for training, validation, and testing, followed by testing accuracy, and testing area under the receiver operating characteristic curve.

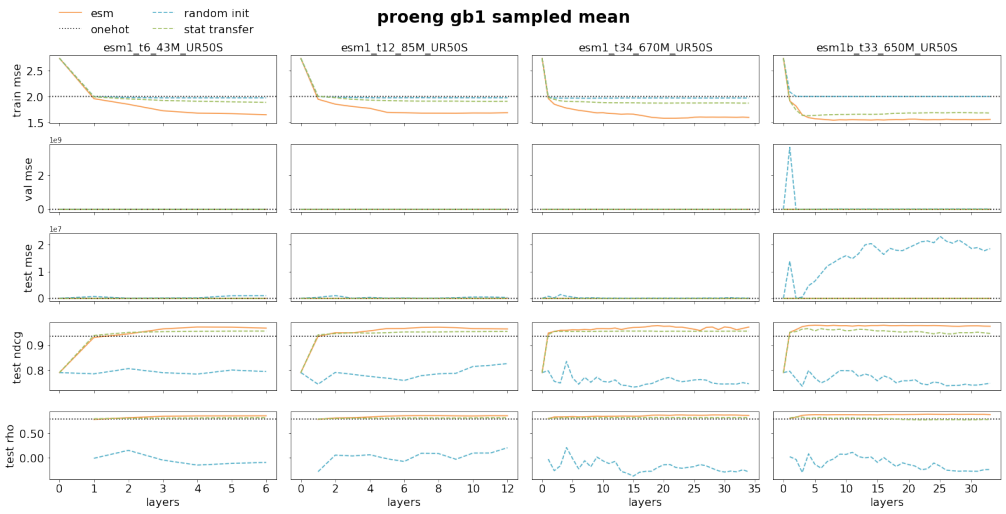


(a) ESM AAV – two vs many out-of-distribution

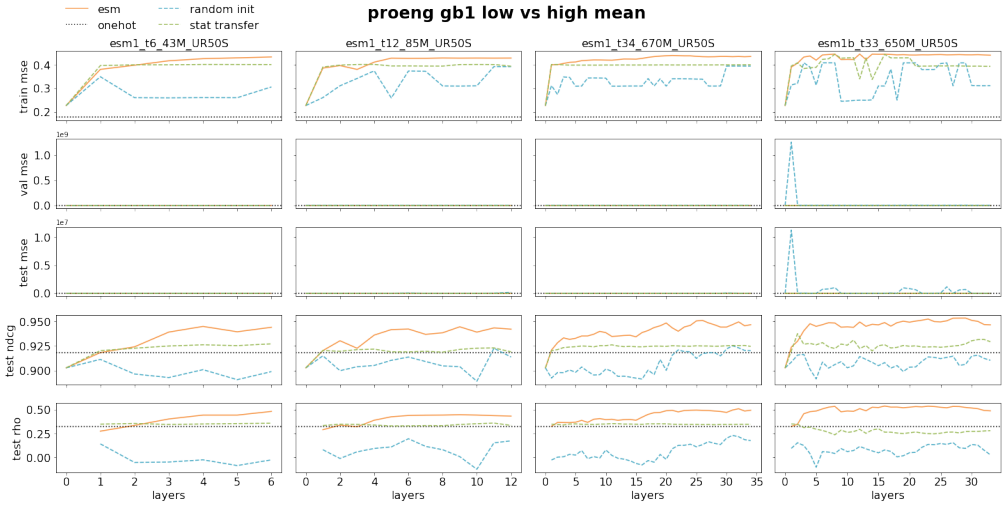


(b) ESM AAV – one vs many out-of-distribution with fewer training samples

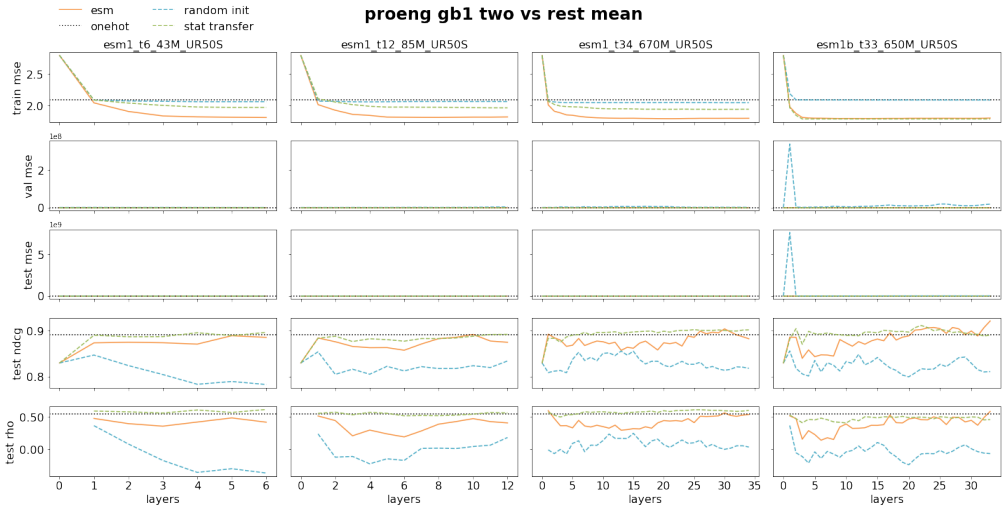
Figure A3: pretrained ESM layer by layer evaluation on two AAV protein engineering datasets



(a) ESM GB1 – sampled



(b) ESM GB1 – low vs high out-of-distribution



(c) ESM GB1 – two vs rest out-of-distribution with fewer training samples

Figure A4: pretrained ESM layer by layer evaluation on three GB1 protein engineering datasets

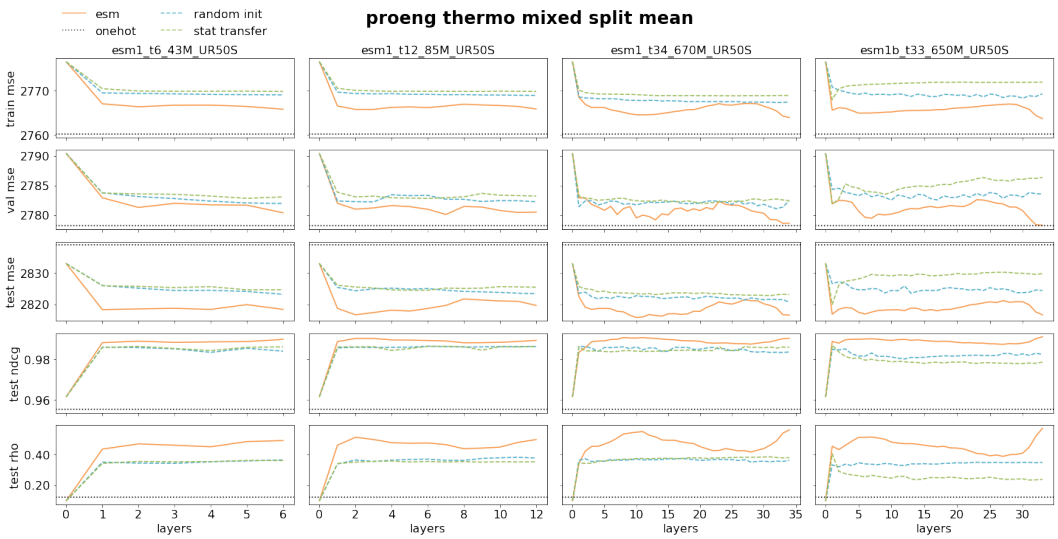


Figure A5: pretrained ESM layer by layer evaluation on a thermostability dataset

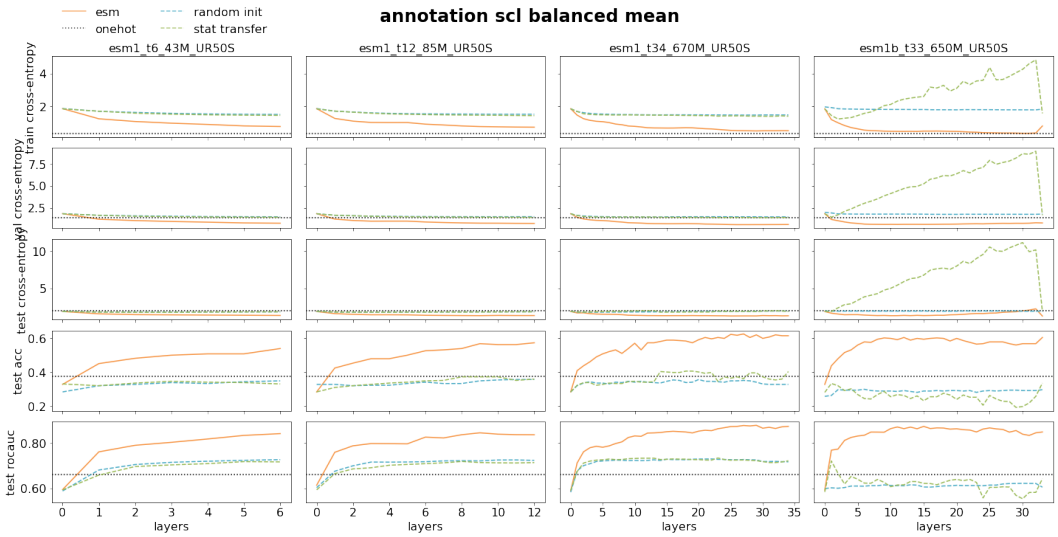
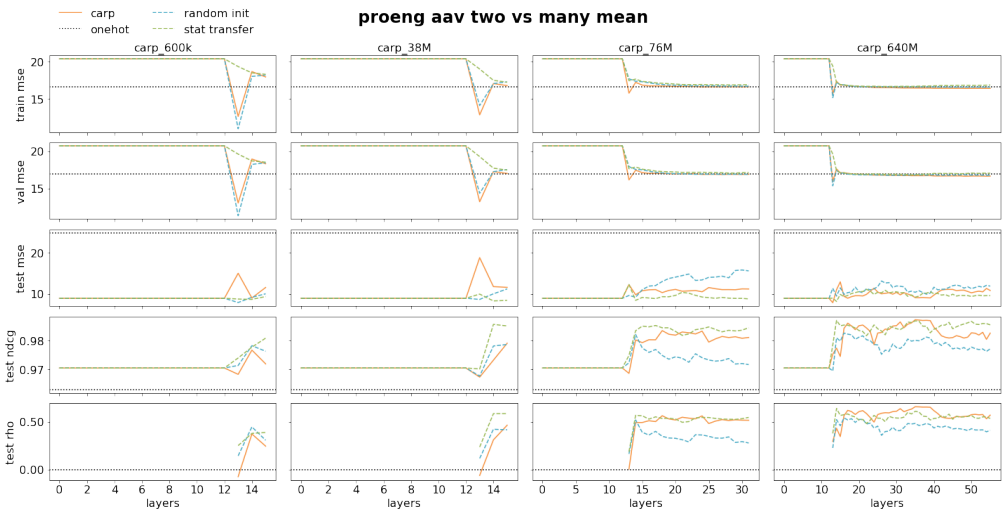
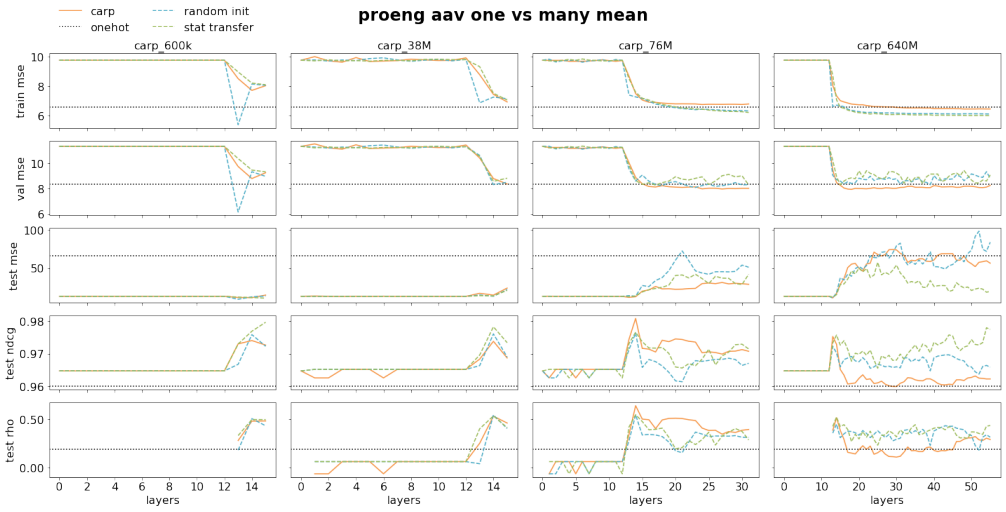


Figure A6: pretrained ESM layer by layer evaluation on an annotation dataset for subcellular localization classification task

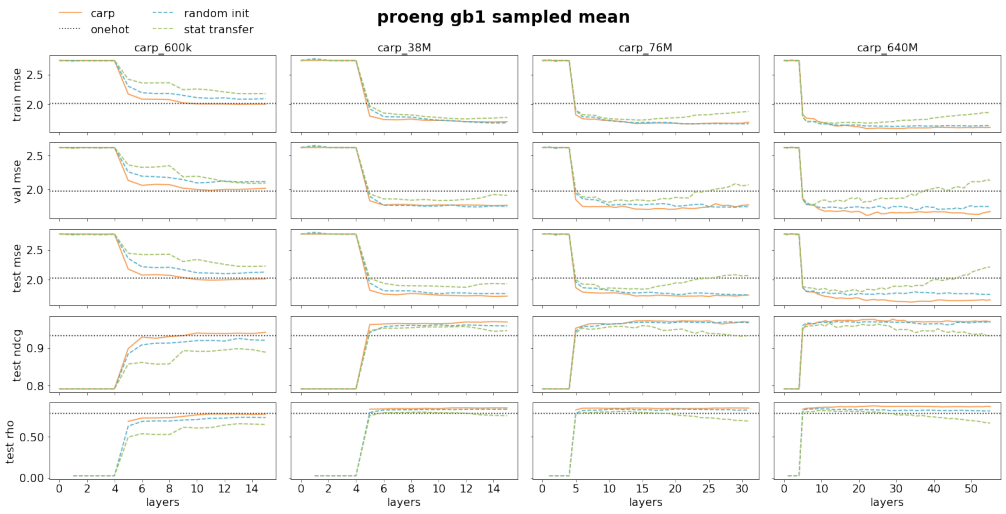


(a) CARP AAV – two vs many out-of-distribution

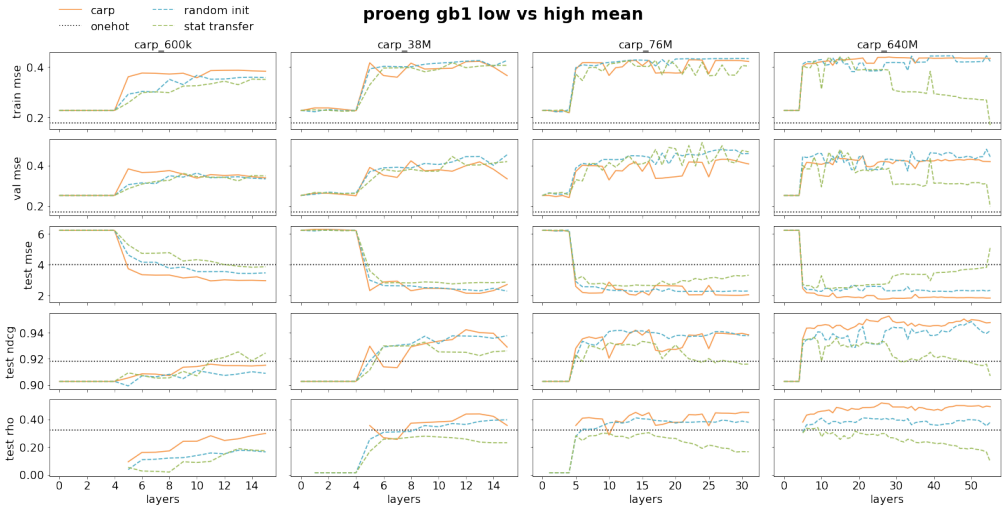


(b) CARP AAV – one vs many out-of-distribution with fewer training samples

Figure A7: pretrained CARP layer by layer evaluation on two AAV protein engineering datasets



(a) CARP GB1 – sampled



(b) CARP GB1 – low vs high out-of-distribution

(c) CARP GB1 – two vs rest out-of-distribution with fewer training samples

Figure A8: pretrained CARP layer by layer evaluation on three GB1 protein engineering datasets

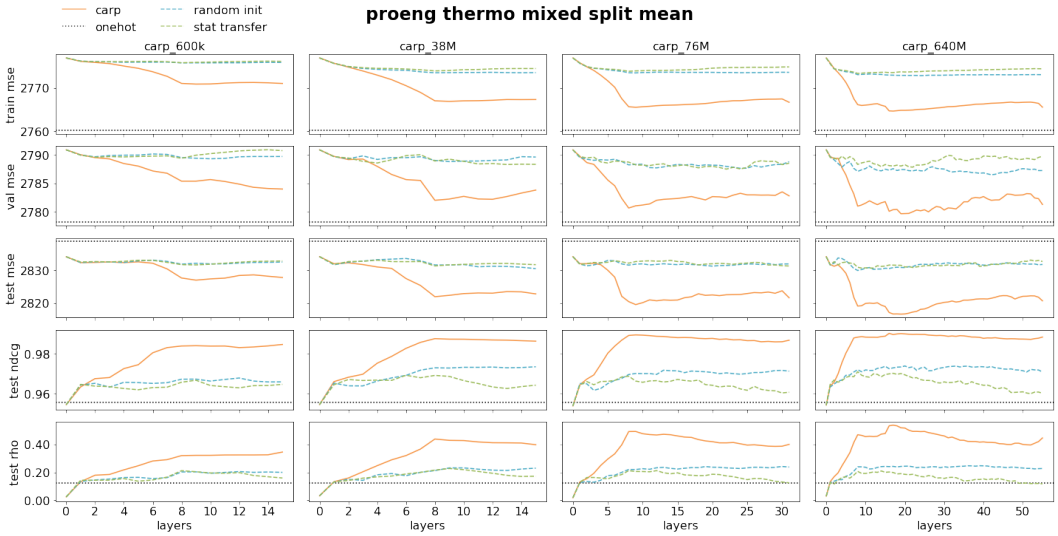


Figure A9: pretrained CARP layer by layer evaluation on a thermostability dataset

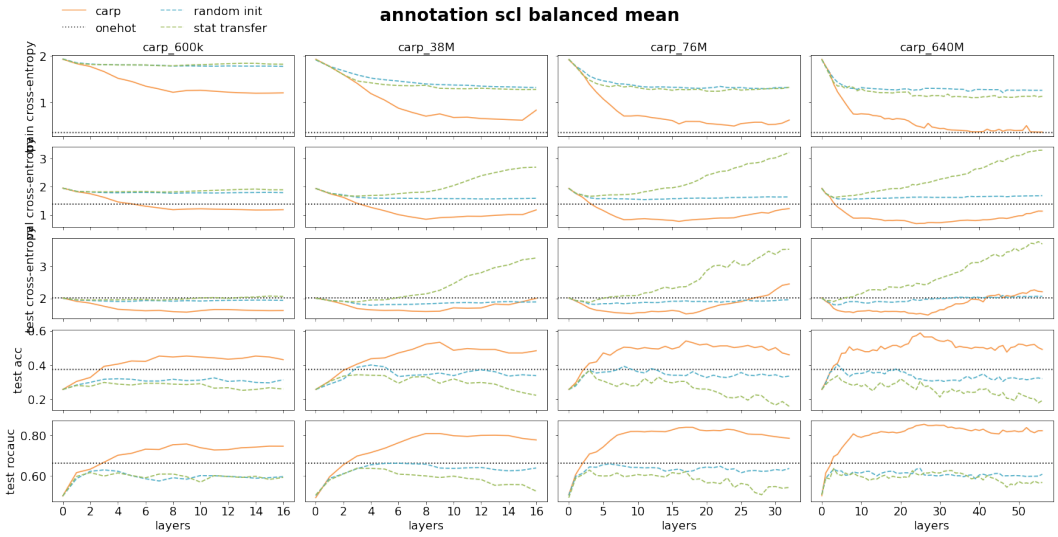


Figure A10: pretrained CARP layer by layer evaluation on a subcellcular classification dataset