CONVOLUTIONS ARE COMPETITIVE WITH TRANSFORMERS FOR PROTEIN SEQUENCE PRETRAINING

Kevin K. Yang, Nicolo Fusi & Alex X. Lu
Microsoft Research New England
Cambridge, MA, USA
{yang.kevin, nfusi, lualex}@microsoft.com

ABSTRACT

Pretrained protein sequence language models have been shown to improve the performance of many functional and structural prediction tasks, and are now routinely integrated into bioinformatics tools. However, these models largely rely on the Transformer architecture, which scales quadratically with sequence length in both run-time and memory. As a result, even state-of-the-art models have limitations on maximum sequence length, limiting what protein sequences can be analyzed with protein language models. To address this limitation, we investigated if more efficient convolutional neural network (CNN) architectures, which scale linearly with sequence length, could be equally as effective as pretrained protein language models. We show that with the masked language model pretraining task, CNNs are competitive to and occasionally superior to Transformers across an extensive set of downstream applications, including structure prediction, zero-shot mutation effect prediction, and out-of-domain generalization. We also demonstrate strong performance on sequences longer than those allowed in the current state-of-the-art Transformer models. Our work has important implications for applications built on protein language models, suggesting that computational efficiency can be improved without sacrificing performance simply by using a CNN architecture instead of a Transformer, and emphasizes the importance of disentangling pretraining task and model architecture in future work studying these models.

1 INTRODUCTION

Large pretrained protein language models have advanced the ability of machine learning models to predict protein structure and function from sequence. These models address the limitation that effective deep learning models generally require an abundance of labeled data to train. Since high-quality labels are only available for a limited number of sequences in most applications, protein language models first expose models to a large quantity of *unlabeled* sequences in a *pretraining* phase (Figure 1a), with the goal of imparting the model with a general foundation of knowledge about protein sequences so that they can be rapidly specialized to downstream tasks of interest with less training data than training from scratch (Figure 1b).

While extensive effort has gone into validating protein language models for downstream applications, relatively little attention has been paid to the architecture of these models. Most works use a Transformer architecture, because these are the architectures employed in analogous work in natural language processing. However, Transformers have numerous drawbacks. First, the compute and memory requirements of these models scale quadratically with input sequence length. In principle, this is more of an issue in pretraining since back-propagating gradients requires additional memory, so longer sequences could be used during evaluation than during pretraining. However, a second limitation of Transformers is that because attention modules (Figure 1c) in these models are invariant to position, the position of each amino acid in a sequence must be encoded as part of the input. In most formulations, this position encoding is difficult to extend past the maximum length seen during training, and as a result, most popular pretrained language models limit the input length of sequences during both pretraining and inference. For example, ESM-1b and ESM-1v have a maximum length of 1022 residues. Unfortunately, this excludes many proteins of interest: of the 42 million cluster

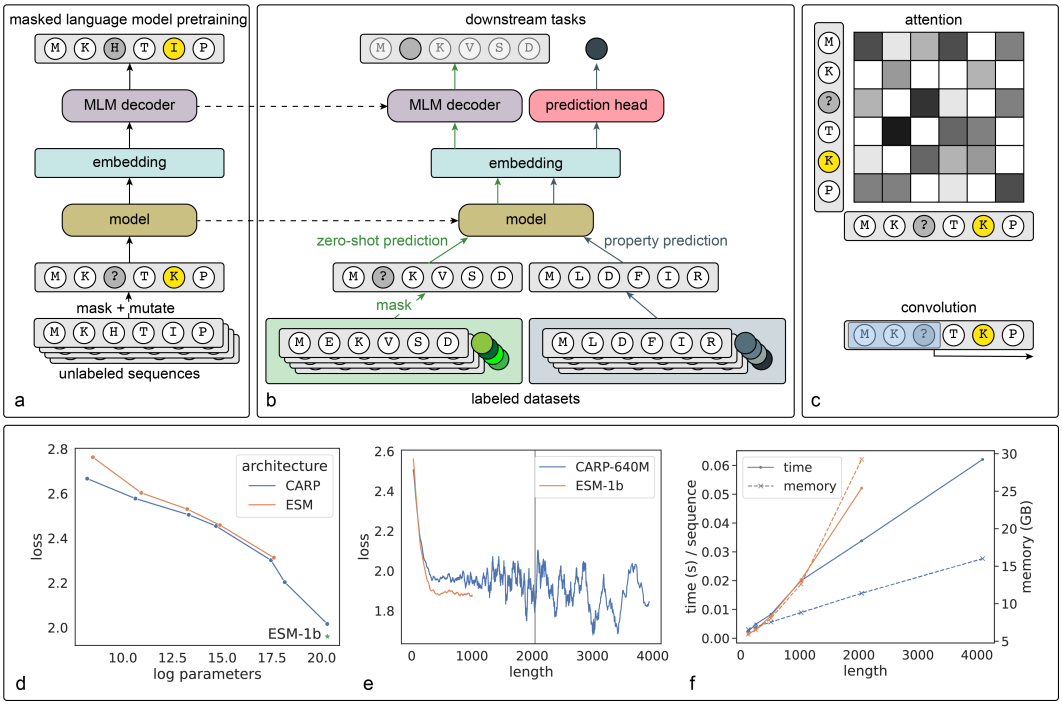


Figure 1: a) Pretraining phase. Models are pretrained with masked language modeling: amino acids in unlabeled protein sequences are randomly masked and mutated to other amino acids, and the model is trained to recover the original sequence. b) After pretraining, models are specialized to downstream tasks. The weights of the pretrained model are transferred to the new task (dotted lines). For some zero-shot tasks like predicting the impact of mutations, the masked language modeling decoder is also useful and can be transferred. Otherwise, the decoder is replaced with a prediction head trained to output a prediction useful for the downstream task. c) Visual explanation of architecture options. Transformers parse sequences through self-attention modules (top), which are $L \times L$ matrices (requiring quadratic time and memory). CARP instead uses convolutions (bottom), which parse sequences with sliding window blocks, scaling linearly with sequence length L. d) Test loss on masked language pretraining task against number of parameters in model for CARP (blue) versus the state-of-the-art Transformer models (ESM - orange). The green asterisk shows the previously reported training loss of ESM-1b on their test dataset (which we did not retrain on our own dataset). e) Test loss on the masked language pretraining task for CARP-640 (blue) and ESM-1b against length of protein sequences. ESB-1b accepts a maximum of 1022-length sequences. CARP-640M is trained on a maximum of 2048 length sequences (vertical grey line), but can extend to arbitrarily long sequences during test time (we test up to length 4096). f) Runtime and memory for forward passes with different input sequence lengths for the CARP-640M (blue) and ESM-1b (orange) architectures, as measured using PyTorch (Paszke et al., 2019) automatic mixed precision on a 48GB Nvidia A100 GPU.

representatives in the March 2020 release of UniRef50 (Suzek et al., 2015), 1.1 million, or 2.6%, are longer than 1022 residues, including the SARS-Cov-2 spike glycoprotein and the *Streptococcus pyogenes* CRISPR-associated endonuclease Cas9.

We reasoned that exploring alternative architectures could improve the computational efficiency of protein language models while allowing for a greater range of (longer) sequences to be studied. In this paper, we study if convolutional neural networks (CNNs) can be effective as pretrained protein language models. Unlike Transformers, CNNs scale linearly with input sequence size. Moreover, CNNs inherently incorporate relative positional information, since sequences are modeled as sliding windows of amino acids (rather than amino acids being treated as independent tokens in the Transformer framework) (Figure 1c). While no previous works have employed CNNs as large pretrained protein language models, prior studies have shown that CNNs are effective at predicting

variant fitness for single protein families (Shin et al., 2021), annotating protein function Bileschi et al. (2022), and in smaller-scale methods design studies (Lu et al., 2020), supporting our hypothesis that CNNs are effective for protein sequences.

We trained protein sequence CNN masked language models, which we refer to as CARP (**C**onvolutional **A**utoencoding **R**epresentations of **P**roteins). We show that CARP models are competitive with the current state-of-the-art Transformer model ESM (Rives et al., 2021; Meier et al., 2021) on a variety of downstream prediction tasks, including structure prediction, zero-shot mutation effect prediction, and out-of-domain generalization on biologically-relevant protein engineering datasets. Because CARP scales linearly in computation with the input sequence and does not rely on an input positional embedding, it is straightforward to apply it to sequences longer than the longest sequences in training, which we demonstrate with zero-shot predictions of mutation effects in CRISPR-Cas9. Overall, the result that CNN models can perform as well as Transformers on downstream tasks has profound implications as these models are increasingly adopted for bioinformatics applications: our results suggest that tools built on large language models can improve their computational efficiency without sacrificing performance, simply by using a model with our CNN architecture over a Transformer. To faciliate these uses, we open-source and release weights and code for our CARP models.

2 Results

2.1 CARP achieves similar performance on the pretraining task to ESM

We trained a series of 7 CARP models with increasing numbers of parameters on nearly 42 million sequences from UniRef50 (see Supplementary Table S1 for details on parameters and architecture). Our largest CARP, CARP-640M, contains approximately 640M learnable parameters, comparable with the popular Transformer model ESM-1b's 650 million parameters.

First, we sought to understand if our CARP architectures could solve masked language modeling as well as Transformer architectures. In this set-up, we are not yet assessing the effectiveness of these models on downstream function or structure prediction tasks, only how well the models can solve the pretraining task of predicting randomly masked or mutated amino acids in unseen sequences. There is no guarantee that performing well on the pretraining task necessarily translates to better performance on all downstream tasks of interest, as this requires that features that these models learn to extract from protein sequences to solve the pretraining task also be features useful to downstream tasks. However, at minimum, performing well on the pretraining task suggests that the models are learning effectively from the pretraining task. For example, if CNNs underperform Transformers on the pretraining task, this would suggest that CNNs are less effective at learning features from masked language modeling - we sought to rule out that this was the case.

We compared the loss of our CARP models against ESM Transformer models with similar numbers of parameters (Figure 1d) on our held-out test dataset. We note that ESM-1b trained and tested models on an earlier version of UniRef50 with with different train/test splits than our work. To produce a more fair comparison, we re-trained some ESM models using our dataset (orange line in Figure 1d), but because reproducing their largest model is too computationally expensive, we simply report results from ESM-1b on its test dataset (green asterisk in Figure 1d).

Overall, we observe that CARP's performance on the pretraining task is comparable to ESM's across several orders of magnitude of variation in the number of parameters when using the same pretraining dataset. Our largest model, CARP-640M, has a test loss of 2.02, comparable to ESM-1b's loss of 1.96 on its test set.

Next, we asked how performance on the pretraining task interacts with sequence length. In principle, one advantage of CNNs is that they can scale to arbitrarily long sequences (including sequences longer than those seen during training), but for this advantage to be practical, the model must generalize to longer sequences. Figure 1e shows the masked language modeling loss by length for CARP-640M and ESM-1b on their respective test sets, smoothed with a window of 30 in the length dimension. For both models, the pretraining loss improves quickly until the sequence length reaches about 500 (shorter sequences are generally more challenging because they provide less context to reconstruct masked tokens), and then slowly thereafter. The maximum input length for ESM-1b during training

is 1022, and this cannot be extended during test time. In contrast, for CARP-640M the maximum input length during training is 2048, but we calculate test losses for sequences with up to 4096 residues. Our results indicate that sequences with a length greater than 2048 have a comparable loss to sequences between 500-2048 length, suggesting that CARP-640M generalizes the pretraining task to sequences longer than those seen during training.

2.2 CARP'S RUN-TIME AND MEMORY SCALE LINEARLY WITH SEQUENCE LENGTH

Next, to confirm that our CARP models are more computationally efficient, we compared the run-time and memory requirements of the CARP-640M architecture against ESM-1b's architecture at different sequence lengths (Figure 1f). Because the original ESM-1b model caps sequence lengths at 1022 residues, we randomly initialize a new ESM-1b model with a longer positional embedding (4096 residues). Although this new model is not expected to encode useful information for transfer to downstream tasks because it has not been pretrained, it serves as a good estimate of computational performance since run-time and memory is generally determined by the number of parameters and architecture, not the specific values of the parameters.

In general, we observe that while for smaller sequences, CARP-640M and ESM-1b have similar run-time (up to 1024 residues) and memory (up to 512 residues) requirements, ESM-1b scales both quadratically while CARP-640M scales linearly. As a result, due to memory limitations in our GPU, we are only able to evaluate sequences around 2048 residues with ESM-1b before an OOM (out of memory) error, while sequences of 4096 residues (and longer) are still possible on our GPU set-up with CARP-640M.

2.3 CARP ACHIEVES COMPARABLE PERFORMANCE ON DOWNSTREAM TASKS TO ESM

A key goal of protein language models is to encode information that can be rapidly transferred to improve performance on downstream prediction tasks (Figure 1b). Depending on the task, different methods for adapting protein language models are appropriate. *Zero-shot* methods do not require access to labels for further training, simply interacting with the pretrained model as is, and are well-suited for tasks where labels are too scarce to train a new model. Alternatively, if labels for a training dataset are available, a small neural network (a "prediction head") can be built on top of representations from the pretrained model and trained to predict these labels. When training the prediction head, the pretrained model can be *frozen*, meaning that its parameters are not allowed to change and only the prediction head is trained, or *fine-tuned*, meaning the pretrained model's parameters are trained jointly with the prediction head's. Often, the decision of whether to freeze or fine-tune the pretrained model depends on how much training data is available: because more parameters are trainable when fine-tuning, there is a greater risk of overfitting.

To assess if CARP is capable of improving performance on downstream tasks, we curated a wide range of benchmarks, including predicting structure, the impact of mutations on fitness, and functional properties such as fluorescence, stability, or melting temperature. When training with downstream labels, we evaluate both freezing ("pt-fr") and fine-tuning ("pt-ft") our pretrained model, and compare against ESM Transformer models. Finally, to confirm that downstream performance is improved by the masked language modeling task, we provide a variety of baselines that do not use pretraining, including randomly-initialized weights ("na-fr" and "na-ft" in our tables), and linear ridge regression and a small CNN built on one-hot amino acid encodings of the protein sequences.

2.3.1 PROTEIN STRUCTURE

One of the most striking successes of protein language models is their ability to encode structural information without access to structural labels during pretraining. We evaluate CARP-640M's ability to encode structural information through 3 tasks:

1. **Remote contact prediction** asks a model to predict whether the C_β atoms of two residues separated by at least 24 residues in the primary structure are within 8 Angstroms of other in the three-dimensional structure. We train on the trRosetta (Yang et al., 2020) training set and evaluate the precision of the top L predictions on the CAMEO hard (Haas et al., 2018) and CASP13-FM (Shrestha et al., 2019) test sets. For contact prediction, we downsample CARP embeddings to 128 dimensions, perform an outer product to produce 2-dimensional

embeddings, and then pass that to a 24-layer dilated residual CNN based on the trRosetta architecture. This is the same as the procedure used by ESM-1b.

2. **Remote homology detection** asks a model to detect structural similarity across distantly-related sequences. We evaluate accuracy on the fold-level holdout set from TAPE.
3. **3-class secondary structure prediction** asks a model to predict whether each residue in a protein is part of a helix, strand, or other. We use the training and validation sets from TAPE and evaluate accuracy on the CB513 test set. For this task, we train a neural network consisting of two CNN layers, an LSTM, and a linear head on top of the pretrained model, as described in Rives et al. (2021).

Since the structural tasks are computationally expensive, we do not fine-tune pretrained models for these tasks (and only train on top of frozen models), and only provide metrics for ESM-1b previously available in literature.

As shown in Table 1, pretraining improves performance for structure prediction tasks, and CARP-640M is competitive with ESM-1b. These results show that pretrained convolutions learn structural information from single sequences, just as pretrained transformers do.

Table 1: Structure prediction tasks. Values for ESM-1b on CASP-13 and CAMEO are taken from Rives et al. (2021). Uncertainties are standard deviations on 3 replicates with different weight initializations.

		Task			
Method	Model	CASP-13 FM	CAMEO	remote homology	secondary structure
pt-fr	CARP-640M	23.7	42.0	0.24±0.008	**0.83**±0.001
	ESM-1b	**28.2**	**44.4**	0.22±0.02	0.81±0.007
pt-ft	CARP-640M	-	-	**0.28**±0.008	**0.83**±0.001
	ESM-1b	-	-	**0.27**±0.006	0.81±0.002
na-fr	CARP-640M	9.7	12.6	0.09±0.02	0.65±0.02
	ESM-1b	-	-	0.09±0.009	0.64±0.002
na-ft	CARP-640M	-	-	0.09 ± 0.02	0.71±0.0005
	ESM-1b	-	-	0.08±0.006	0.70±0.002

2.3.2 Zero-shot mutation effect prediction

Pretrained protein language models can predict experimental measurements of protein function without further training on sequence-fitness measurements or sets of evolutionarily-related sequences Hie et al. (2022); Meier et al. (2021). Following Meier et al. (2021), we score CARP-640M on 41 deep mutational scanning datasets originally compiled by Riesselman et al. (2018). These datasets measure the effects of thousands of mutations or combinations of mutations to a parent sequence. Details are described in Section 4.4.

Figure 2 compares zero-shot performance for CARP-640M, ESM-1b, ESM-1v, position-specific scoring matrices (PSSM), and ProtBert-BFD. ESM-1v results are for an ensemble of five transformers pretrained on UniRef90. Averaged across the 41 datasets, CARP-640M has a Spearman correlation of 0.49, compared to 0.46 for ESM-1b, 0.51 for ESM-1v, 0.46 for PSSM, and 0.43 for ProtBERT-BFD. CARP-640M outperforms ESM-1b on 22 out of 41 datasets, ESM-1v on 18 out of 41 datasets, PSSM on 26 out of 41 datasets, and ProtBERT-BFD on 25 out of 41 datasets.

Meier et al. (2021) found that using the full UniProt sequences instead of only the sequence of the mutated domain results in better zero-shot predictions. However, this is not always possible with ESM-1x, as some UniProt sequences for these proteins are longer than 1022 residues. As a further proof of concept, we made zero-shot predictions for the effects of mutations in Cas9 from *Streptococcus pyogenes* (Spencer and Zhang, 2017), which is 1368 residues long, and obtain a Spearman correlation of 0.26. These results show that pretrained convolutions can make zero-shot predictions of protein mutation effects on fitness, including on sequences longer than allowed by ESM-1x.

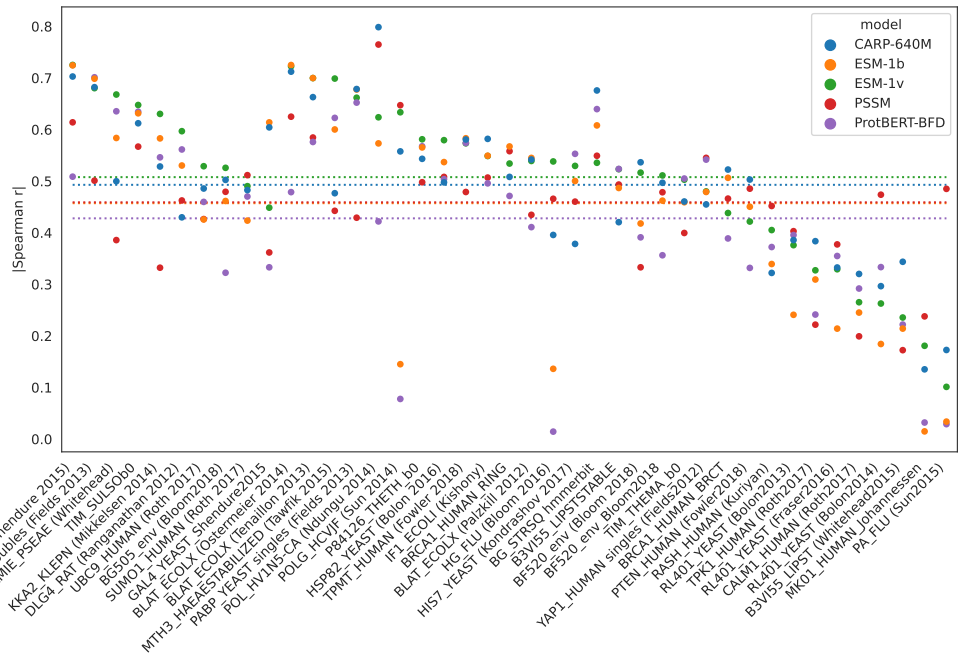


Figure 2: Zero-shot protein fitness prediction. Comparison across 41 deep mutational scanning datasets from DeepSequence. Points are Spearman correlation on each dataset. Horizontal lines show the average Spearman correlation across the datasets for each model. Values for ESM-1b, ESM-1v, PSSM, and ProtBERT-BFD are taken from Meier et al. (2021).

2.3.3 Out-of-domain fitness prediction

Another motivation for pretrained protein language models is that because they build on a general foundation of knowledge about proteins learned in the pretraining phase, they may be able to better extrapolate from limited training data. For example, a protein engineer may want to train a model on single mutants and make predictions for sequences with multiple mutations, or train a model that is accurate for sequences with fitness greater than what is seen in the training set. To test these kinds of out-of-domain prediction schemes, we evaluate CARP-640M on tasks from the following landscapes from FLIP (Dallago et al., 2021).

1. AAV (Table 2): Adeno-associated virus (AAV) capsid proteins are responsible for helping the virus integrate a DNA payload into a target cell (Vandenberghe et al., 2009), and there is great interest in engineering versions of these proteins for gene therapy (Büning et al., 2015; Barnes et al., 2019). Bryant et al. (2021) measure a rich mutational screening landscape of different VP-1 AAV proteins.
2. GB1 (Table 3): GB1 is the binding domain of protein G, an immunoglobulin binding protein found in Streptococcal bacteria. In their original study, Wu et al. (2016) measured the fitness of 149,361 out of 160,000 possible combinations of mutations at 4 positions.

For each landscape, we evaluate several data splits where test sequences differ from training sequences:

- x-vs-many: Train on sequences with up to x mutations and test on the remainder of the landscape.
- mut-des: Train on sequences sampled from mutagenesis libraries and test on sequences designed by machine-learning models (AAV only).
- low-vs-high: Train on sequences with fitnesses below the wild-type and test on sequences with fitnesses above the wild-type.

Table 2: Performance on the FLIP AAV tasks. Values for models other than CARP-640M are taken from Dallago et al. (2021). Uncertainties for ESM-1b and CNN are standard deviations over 10 random seeds. Uncertainties for CARP-640M are standard deviations over 3 random seeds. Dallago et al. (2021) do not provide uncertainties for the mut-des task because of the computational cost.

		Task				
Method	Model	1-vs-many	2-vs-many	7-vs-many	mut-des	low-vs-high
pt-fr	CARP-640M	0.31±0.18	0.51±0.18	0.58±0.14	0.75±0.08	0.25±0.09
	ESM-1b	0.03±0.11	0.61±0.04	0.65±0.01	0.76	**0.38**±0.01
pt-ft	CARP-640M	**0.73**±0.05	**0.81**±0.03	**0.77**±0.03	**0.85**±0.003	0.19±0.08
na-fr	CARP-640M	0.48±0.07	0.50±0.05	0.60±0.05	0.76±0.02	0.21±0.02
	ESM-1b	0.18±0.01	0.20±0.03	0.38±0.04	0.56	0.06±0.01
na-fr	CARP-640M	0.04±0.12.	0.50±0.43	0.38±0.37	0.84±0.01	0.24±0.21
baseline	ridge	0.22	0.03	0.65	0.68	0.12
	CNN	0.35±0.11	0.58±0.09	0.73±0.004	0.71	0.28±0.02

In general, pretraining improves CARP-640M's performance on these tasks, and fine-tuning the entire model outperforms freezing the pretrained weights. Comparisons to the baselines show that pretraining is most helpful when generalizing from single mutants to multiple. When not fine-tuning all the way through, there is little benefit from pretraining, and on some tasks pretraining hurts performance. CARP-640M outperforms ESM-1b on generalizing from few mutations to more, but ESM-1b is better at generalizing from a low-fitness training to higher-fitness sequences. These results show that pretrained convolutions help generalization to types of sequence variation not seen during training. On GB1, finetuning ESM-1b end-to-end instead of freezing the pretrained weights hurts its performance, while CARP-640M benefits from full finetuning. In addition, CARP-640M provides better representations without pretraining than ESM-1b on all the AAV tasks and 2 of the 4 GB1 tasks, showing that the architecture alone also influences generalization.

Table 3: Performance (Spearman correlation) on the FLIP GB1 tasks. Values for models other than CARP-640M and ESM-1b with full finetuning are taken from FLIP. Uncertainties for ESM-1b frozen and CNN are standard deviations over 10 random seeds. Uncertainties for CARP-640M and ESM-1b with full finetuning are standard deviations over 3 random seeds.

		Task			
Method	Model	1-vs-many	2-vs-many	3-vs-many	low-vs-high
pt-fr	CARP-640M	0.15±0.18	0.18±0.23	0.62±0.06	0.12±0.03
	ESM-1b	**0.29**±0.02	0.47±0.05	0.79±0.01	**0.53**±0.03
pt-ft	CARP-640M	0.19±0.26	**0.73**±0.03	**0.87**±0.004	0.43±0.04
	ESM-1b	0.11±0.11	0.67±0.07	0.66±0.18	0.42±0.09
na-fr	CARP-640M	0.03±0.03	0.07±0.17	0.71±0.03	0.35±0.03
	ESM-1b	0.12±0.01	0.21±0.01	0.52±0.01	0.32±0.03
na-ft	CARP-640M	0.11±0.07	0.38±0.26	0.68±0.33	0.23±0.26
	ESM-1b	0.05±0.28	0.14±0.13	0.10±0.13	-0.04±0.09
baseline	ridge	0.28	0.59	0.76	0.34
	CNN	0.15±0.09	0.39±0.04	0.81±0.004	0.47±0.01

2.3.4 In-domain property prediction

Finally, we consider property and fitness prediction tasks that do not require difficult biological generalization (Table 4). We evaluate on three sequence-fitness regression tasks:

1. **Fluorescence** requires the model to predict the effect of one or more mutations on the brightness of green fluorescent protein. The data was originally collected by Sarkisyan et al. (2016). We use the data splits provided in TAPE.
2. **Stability** requires the model to predict a small protein's resistance to protease degradation. The data was originally collected by Rocklin et al. (2017). We use the data splits provided in TAPE.
3. **Meltome-mixed** requires the model to predict the melting temperature of a variety of proteins from across the domains of life. The data was originally collected by Jarzab et al. (2020). We use the cluster representatives and data splits provided in FLIP.

In addition, we evaluate on two intrinsically disordered region (IDR) function classification tasks taken from Zarin et al. (2021). For the IDR datasets, we use MMseqs2 (Steinegger and Söding, 2017) to cluster sequences to 50% identity and then randomly assign clusters to training, validation, or testing.

1. **Cdc28 binding** requires the model to predict whether an IDR is a target of Cdc28.
2. **Mitochondria targeting** requires the model to predict whether an IDR targets its protein for transport into the mitochondria.

Table 4: Performance on in-domain tasks. For fluorescence, stability, and meltome, values reported are Spearman correlation. For the IDR tasks, values reported are area under the reciever operating curve. Values for ESM-1b on fluorescence and stability are taken from Rives et al. (2021). Values for baselines on fluorescence and stability are taken from FLIP. Uncertainties for ESM-1b and CNN are standard deviations over 10 random seeds except on the IDR tasks, where they are over 3 random seeds. Uncertainties for CARP-640M are standard deviations over 3 random seeds. We do not calculate uncertainties on meltome due to the computational cost.

		Task				
Method	Model	fluorescence	stability	meltome	Cdc28	mito.
pt-fr	CARP-640M	0.58±0.02	0.62±0.03	0.54	0.84±0.01	0.86±0.02
	ESM-1b	-	-	**0.67**±0.01	**0.91**±0.004	**0.90**±0.01
pt-ft	CARP-640M	**0.68**±0.002	**0.72**±0.01	0.53	0.88±0.02	0.89±0.004
	ESM-1b	**0.68**	0.71	-	0.89±0.01	0.88±0.01
na-fr	CARP-640M	0.62±0.01	0.52±0.17	0.29	0.84±0.01	0.86±0.01
	ESM-1b	-	-	0.45±0.03	0.88±0.01	0.84±0.03
na-ft	CARP-640M	0.58±0.07	0.65±0.05	0.30	0.79±0.03	0.87±0.01
	ESM-1b	-	-	-	0.83±0.02	0.85±0.02
	ridge	**0.68**	0.48	0.17	0.52	0.53
	CNN	0.67	0.51	0.34±0.01	0.84±0.02	0.87±0.02

In general, while pretraining improves CARP-640M's performance on these tasks, neither of the large pretrained models consistently out-perform the baseline models on these tasks. Almost all the models perform very well on the IDR tasks, indicating that performance is saturating on these tasks. Nevertheless, CARP-640M is generally comparable to ESM-1b, showing that once again pretrained convolutions are comparable to pretrained attention.

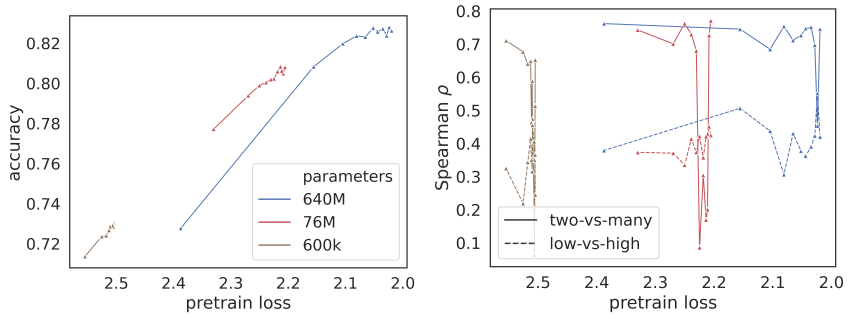
2.4 Evaluating performance at checkpoints illuminates relationship between pretraining and downstream tasks

While we showed that our CNN CARP models perform comparably to ESM models across protein prediction tasks, not all downstream tasks benefit from pretraining with the masked language model task, with simple regression baselines performing as well as both CARP and ESM models in some tasks. One possibility is that for some tasks, the features learned to solve the masked language modeling pretraining task may not overlap or contain any useful information for some downstream tasks. We reasoned that one way to test this hypothesis would be to evaluate performance on

downstream tasks at various checkpoints of our CARP models: if pretraining does learn features useful for downstream tasks, we expect a relationship where better performance on the masked language modeling task (which improves as models are pretrained for longer) has at least some correlation with performance on the downstream tasks.

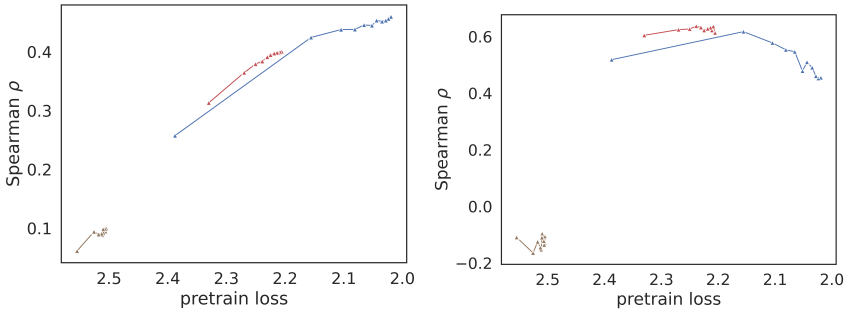
We finetuned pretraining checkpoints for CARP-600k, CARP-76M, and CARP-640M on secondary structure, remote homology, and the FLIP GB1, AAV, and meltome tasks. Figures 3a and S2a show that structure prediction improves smoothly as the model size increases and the model is pretrained longer. This confirms that, as for transformers, pretraining imparts structural information to CNNs. However, Figures 3b, S2b, and S2c shows that this relationship does not exist for the out-of-domain FLIP tasks. In many cases, a small amount of pretraining is sufficient to outperform the naive baseline, and further pretraining has an unpredictable and often negative effect on performance. Finally, for the FLIP meltome task, Figure S2d shows that performance generally improves as CARP is pretrained, but the pretraining effect saturates, and CARP-76M outperforms CARP-640M.

Figure 4a shows that the average zero-shot performance improves with both model size and pretraining performance. However, this is not the case for every individual dataset within DeepSequence. Figure 4b shows a case where zero-shot performance peaks and then declines as CARP is pretrained. The Spearman correlation between the pretrain loss and zero-shot Spearman correlation range from 1 (monotonic increase in zero-shot performance with more pretraining) and -0.9, as shown in Figure S1. The average over the DeepSequence datasets is 0.40 for CARP-640M, 0.48 for CARP-76M, and 0.23 for CARP-600k. Although CARP-640M has better overall zero-shot performance than CARP-76M, CARP-76M more consistently improves with more pretraining than CARP-640M. The heterogeneity in the relationship between pretraining performance and zero-shot performance suggests that many but not all zero-shot tasks in DeepSequence are strongly determined by structural stability.



(a) Secondary structure (b) GB1

Figure 3: Effect of model size and checkpoint pretrain loss on downstream performance.



(a) DeepSequence-mean (b) DMS YAP1 human

Figure 4: Effect of model size and checkpoint pretrain loss on zero-shot performance.

3 Conclusion and Discussion

We show that CNNs can be comparable to or superior to Transformers on the masked language modeling pretraining task, and that pretraining helps CNNs achieve similar levels of performance as Transformers when adapting these models to downstream protein property prediction tasks. Our results challenge the tightly-held association between masked language modeling and Transformers, and shows that the pretraining task itself, not the Transformer architecture, is the essential component that makes pretraining effective. We believe that this insight is of practical utility to researchers working on proteins especially as protein language models are becoming a workhorse of bioinformatics methods. Unlike Transformers, CNNs scale linearly with input sequence length, which becomes important when modeling long protein sequences. Analogous work in natural language processing has also shown that CNNs can require fewer FLOPs of compute than Transformers, even for short sequences (Tay et al., 2021). In addition, while we use standard dilated convolutions, there are more efficient convolution variants designed for sequence modeling (Wu et al., 2019) that may further improve model speed. Together, our work suggests that CNNs can improve the computational efficiency of prediction methods built on protein language models with no impact on performance, and further advances in these architectures, which are relatively underexplored compared to Transformers, may heighten this improvement.

One limitation of our CNN architecture is that unlike Transformers, it does not use a cross- or self-attention module. In at least some tasks, these modules can contribute towards interpretability. For example, it is possible to extract structural contact maps from pretrained Transformer self-attention matrices (Rao et al., 2020), and self-attention matrices contain information about binding sites (Vig et al., 2020). Convolutions lack an obvious equivalent. In addition, attention-based models naturally extend to predict protein-protein interaction sites, because they provide a ready framework for pairwise interactions across amino acids between sequences. Finally, while we highlighted issues with computational efficiency of Transformers, recent technical advances have sought to address these challenges. For example, the challenge of quadratic dependence on sequence length can be ameliorated with approximate attention methods (Child et al., 2019; Beltagy et al., 2020; Kitaev et al., 2020; Tay et al., 2020a; Wang et al., 2020; Zaheer et al., 2020; Katharopoulos et al., 2020; Choromanski et al., 2020a) (although we note the choice of approximation matters for performance and the best method is not always clear *a priori* (Tay et al., 2020b)). On proteins, Choromanski et al. (2020a) and Choromanski et al. (2020b) show that Performer approximate attention can perform well for autoregressive and masked protein language models, respectively, while ProteinBERT combines a fast global attention mechanism with masked language and functional annotation prediction pretraining (Brandes et al., 2021).

However, we show that without pretraining, CNNs and Transformers perform differently on downstream tasks. This observation suggests that our CNNs may provide complementary inductive biases to those found in Transformer models. Unfortunately, we also find that, while masked language model pretraining is effective in imparting models with structural knowledge, the relationship between model size, pretraining loss, and downstream performance is less stable for out-of-domain protein engineering tasks, indicating that masked language modeling may not be effective for at least some types of tasks and emphasizing a need for more effective pretraining tasks. While we evaluate the effects of masked language model pretraining, numerous other pretraining tasks have been proposed including autoregressive language model pretraining (Madani et al., 2020), pairwise masked language modeling (He et al., 2021), and combining structural information (Mansoor et al., 2021; Zhang et al., 2022; McPartlon et al., 2022; Hsu et al., 2022; Chen et al., 2022; Wang et al., 2022) or functional annotations (Brandes et al., 2021). Together, our work demonstrates the importance of disentangling pretraining task and architecture. We hope that this work is the first step in investigating the independent and interacting effects of pretraining and architecture for protein sequence modeling.

Data, code, and model availability

Model code is available at `https://github.com/microsoft/protein-sequence-models`. Pretrained model weights and our train/validation/test splits for the two IDR datasets and UniRef50 are available at `https://doi.org/10.5281/zenodo.6564798`.

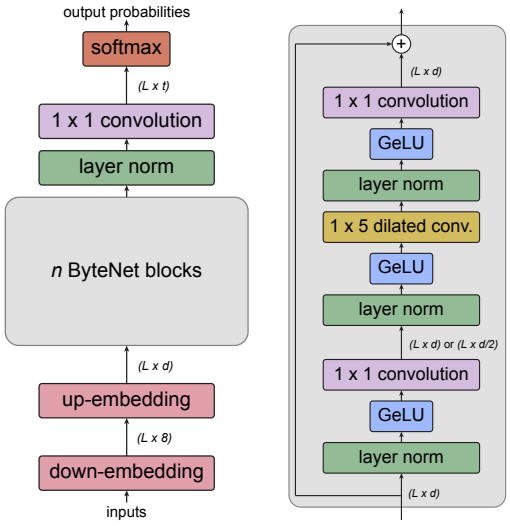
ACKNOWLEDGMENTS

We thank Brian Hie, Roshan Rao, Joshua Meier, and Alexander Rives for assistance with the zero-shot mutation effect prediction datasets.

4 METHODS

4.1 ARCHITECTURE

One limitation of CNNs is that they are locally connected, so neurons in the model may not see the entire sequence at once, preventing the learning of distal interactions. To overcome this limitations, ByteNet uses a dilated CNN architecture Kalchbrenner et al. (2016), which increases the CNN perceptive field exponentially with the number of layers, allowing the model to obtain global context for long input sequences.



(a) CARP (b) ByteNet block

Figure 5: Schematics of the CARP architecture.

CARP combines the ByteNet encoder with simple input embedding and output decoding layers, as shown in Figure 5a. CARP begins with an embedding layer, which maps an input sequence of L tokens $x \in \mathbb{D}^L$ to an 8-dimensional intermediate embedding, followed by a linear mapping into the model dimension d: $e_0 \in \mathbb{R}^{L \times d}$. This passes through a stack of n ByteNet dilated CNN blocks Figure 5b with residual connections in between followed by a final layer norm to produce the encoder representation $e_n \in \mathbb{R}^{L \times d}$, and finally a linear decoder maps this to the $L \times t$ logits, where t is the number of possible tokens. The 1×5 convolution layer in every ByteNet block is dilated and padded to preserve sequence length. The CNN dilation rate doubles every layer up to a maximum rate r (for our experiments $r = 128$). This scheme is repeated multiple times in the network, always starting from a dilation rate of 1.

Throughout this paper, CARP refers to any ByteNet masked language model, while CARP-X refers to the model with approximately X parameters.) For example, our largest model, CARP-640M, has 640M parameters, comparable to the ESM-1b Transformer, which has 650 million parameters. Hyperparameters for different-sized versions of CARP and ESM are found in Tables S1 and S2, respectively.

4.2 DATASET AND MASKED LANGUAGE MODELING

We train CARP on the cluster representatives from the March 2020 release of UniRef50, with approximately 83k sequences held out for validation and another 210k sequences held out for testing,

leaving 41.5 million sequences for training. Models are pretrained using the masked language model objective described in Rives et al. (2021). Each sequence is corrupted by changing some tokens to a special mask token or another amino acid token, and the model is tasked with reconstructing the original sequence. Specifically, 15% of tokens from each sequence are randomly selected. For those 15% of tokens, 80% are replaced by the mask token, 10% are replaced by a randomly-chosen amino acid, and 10% remain unchanged. The model is trained to minimize the cross-entropy loss between its predictions for the selected tokens and the true tokens at those locations.

4.3 Training Details

We varied the number of parameters in CARP from approximately 3000 to 640 million by setting the model dimension d, setting the encoder hidden dimension h_e to either d or $\frac{d}{2}$, and setting the number of layers. All models are trained with the Adam optimizer, a maximum learning rate of 0.001, a linear warmup for 16,000 steps, and dynamic batching to maximize GPU usage. The largest model, CARP-640M, was trained on 128 32GB Nvidia V100 GPUs for 620,000 updates, or approximately 56 days.

To adapt CARP to downstream tasks, we use the output from the final layer norm in Figure 5a as the output representation. Unless otherwise noted, the prediction head consists of a learned attention that converts the output from $L \times d$ to d followed by a 2-layer neural network with hidden size d. For tasks with labels, we evaluate both freezing and fine-tuning CARP and compare to ESM-1b or ESM-1v. We finetune models with a maximum learning rate of 0.0001, a linear warmup over 1000 steps, and early stopping based on the validation set. Finetuning was performed on one 32 GB V100; depending on the task, finetuning took between several minutes to 48 hours. Where relevant, we also compare the CARP architecture with randomly-initialized weights, linear ridge regression, and the small CNN described in Dallago et al. (2021) and Shanehsazzadeh et al. (2020). In our experiments fine-tuning models across checkpoints, we initialize the prediction heads with the same weights across all model checkpoints of the same size, to control for randomness between prediction heads.

4.4 Zero-shot fitness prediction

For one-shot mutation impact prediction, we score sequences by masking every mutated position and computing the log odds ratio between the mutated and wild-type residues at each mutated position, assuming an additive model when a sequence contains multiple mutations:

$$\sum_{p \in P} \log p(x_P^{\text{mt}} | x_{\setminus P}^{\text{wt}}) - \log p(x_P^{\text{wt}} | x_{\setminus P}^{\text{wt}}) \tag{1}$$

where P indicates the mutated positions.

References

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Baris E Suzek, Yuqi Wang, Hongzhan Huang, Peter B McGarvey, Cathy H Wu, and UniProt Consortium. UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):1–11, 2021.

Maxwell L Bileschi, David Belanger, Drew H Bryant, Theo Sanderson, Brandon Carter, D Sculley, Alex Bateman, Mark A DePristo, and Lucy J Colwell. Using deep learning to annotate the protein universe. *Nature Biotechnology*, pages 1–6, 2022.

Amy X Lu, Haoran Zhang, Marzyeh Ghassemi, and Alan Moses. Self-Supervised contrastive learning of protein representations by mutual information maximization. November 2020.

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. *Proc. Natl. Acad. Sci. U. S. A.*, 118(15), April 2021.

Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alexander Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. In M. Ranzato, A. Beygelzimer, K. Nguyen, P.S. Liang, J.W. Vaughan, and Y. Dauphin, editors, *Advances in Neural Information Processing Systems 34*, 2021.

Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020.

Jürgen Haas, Alessandro Barbato, Dario Behringer, Gabriel Studer, Steven Roth, Martino Bertoni, Khaled Mostaguir, Rafal Gumienny, and Torsten Schwede. Continuous Automated Model Evaluation (CAMEO) complementing the critical assessment of structure prediction in CASP12. *Proteins: Structure, Function, and Bioinformatics*, 86:387–398, 2018.

Rojan Shrestha, Eduardo Fajardo, Nelson Gil, Krzysztof Fidelis, Andriy Kryshtafovych, Bohdan Monastyrskyy, and Andras Fiser. Assessing the accuracy of contact predictions in CASP13. *Proteins: Structure, Function, and Bioinformatics*, 87(12):1058–1068, 2019.

Brian L. Hie, Kevin K. Yang, and Peter S. Kim. Evolutionary velocity with protein language models predicts evolutionary dynamics of diverse proteins. *Cell Systems*, 2 2022. ISSN 24054712. doi: 10.1016/j.cels.2022.01.003.

Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature Methods*, 15(10):816–822, 2018.

Jeffrey M Spencer and Xiaoliu Zhang. Deep mutational scanning of *S. pyogenes* Cas9 reveals important functional domains. *Scientific reports*, 7(1):1–14, 2017.

Christian Dallago, Jody Mou, Kadina E Johnston, Bruce Wittmann, Nick Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. FLIP: Benchmark tasks in fitness landscape inference for proteins. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

LH Vandenberghe, JM Wilson, and G Gao. Tailoring the AAV vector capsid for gene therapy. *Gene therapy*, 16(3):311–319, 2009.

Hildegard Büning, Anke Huber, Liang Zhang, Nadja Meumann, and Ulrich Hacker. Engineering the AAV capsid to optimize vector–host-interactions. *Current opinion in pharmacology*, 24:94–104, 2015.

Christopher Barnes, Olivia Scheideler, and David Schaffer. Engineering the AAV capsid to evade immune responses. *Current opinion in biotechnology*, 60:99–103, 2019.

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*, 39(6):691–696, 2021.

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*, 5:e16965, 2016. doi: 10.7554/eLife.16965.

Karen S Sarkisyan, Dmitry A Bolotin, Margarita V Meer, Dinara R Usmanova, Alexander S Mishin, George V Sharonov, Dmitry N Ivankov, Nina G Bozhanova, Mikhail S Baranov, Onuralp Soylemez, Natalya S Bogatyreva, Peter K Vlasov, Evgeny S Egorov, Maria D Logacheva, Alexey S Kondrashov, Dmitry M Chudakov, Ekaterina V Putintseva, Ilgar Z Mamedov, Dan S Tawfik, Konstantin A Lukyanov, and Fyodor A Kondrashov. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397, 2016. doi: 10.1038/nature17995.

Gabriel J Rocklin, Tamuka M Chidyausiku, Inna Goreshnik, Alex Ford, Scott Houliston, Alexander Lemak, Lauren Carter, Rashmi Ravichandran, Vikram K Mulligan, Aaron Chevalier, et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357 (6347):168–175, 2017. doi: 10.1126/science.aan0693.

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.

Taraneh Zarin, Bob Strome, Gang Peng, Iva Pritišanac, Julie D Forman-Kay, and Alan M Moses. Identifying molecular features that are associated with biological function of intrinsically disordered protein regions. *Elife*, 10:e60220, 2021.

Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

Yi Tay, Mostafa Dehghani, Jai Gupta, Dara Bahri, Vamsi Aribandi, Zhen Qin, and Donald Metzler. Are pre-trained convolutions better than pre-trained transformers? *arXiv preprint arXiv:2105.03322*, 2021.

Felix Wu, Angela Fan, Alexei Baevski, Yann N Dauphin, and Michael Auli. Pay less attention with lightweight and dynamic convolutions. *arXiv preprint arXiv:1901.10430*, 2019.

Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. Transformer protein language models are unsupervised structure learners. In *International Conference on Learning Representations*, 2020.

Jesse Vig, Ali Madani, Lav R Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. Bertology meets biology: Interpreting attention in protein language models. *arXiv preprint arXiv:2006.15222*, 2020.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR, 2020a.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020a.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020b.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, David Belanger, Lucy Colwell, et al. Masked language modeling for proteins via linearly scalable long-context transformers. *arXiv preprint arXiv:2006.03555*, 2020b.

Nadav Brandes, Dan Ofer, Yam Peleg, Nadav Rappoport, and Michal Linial. ProteinBERT: A universal deep-learning model of protein sequence and function. *bioRxiv*, 2021.

Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. ProGen: Language modeling for protein generation. *arXiv*, 2020.

Liang He, Shizhuo Zhang, Lijun Wu, Huanhuan Xia, Fusong Ju, He Zhang, Siyuan Liu, Yingce Xia, Jianwei Zhu, Pan Deng, et al. Pre-training co-evolutionary protein representation via a pairwise masked language model. *arXiv preprint arXiv:2110.15527*, 2021.

Sanaa Mansoor, Minkyung Baek, Umesh Madan, and Eric Horvitz. Toward more general embeddings for protein design: Harnessing joint representations of sequence and structure. *bioRxiv*, 2021.

Zuobai Zhang, Minghao Xu, Arian Jamasb, Vijil Chenthamarakshan, Aurelie Lozano, Payel Das, and Jian Tang. Protein structure representation learning by geometric pretraining. 2022.

Matt McPartlon, Ben Lai, and Jinbo Xu. A deep SE (3)-equivariant model for learning inverse protein folding. *bioRxiv*, 2022.

Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. *bioRxiv*, 2022.

Can Chen, Jingbo Zhou, Fan Wang, Xue Liu, and Dejing Dou. Structure-aware protein self-supervised learning. *ArXiv*, abs/2204.04213, 2022.

Zichen Wang, Steven A. Combs, Ryan Brand, Miguel Calvo Rebollar, Panpan Xu, George Price, Nataliya Golovach, Emmanuel Oluwatobi Salawu, Colby Wise, Sri Priya Ponnapalli, and Peter M. Clark. LM-GVP: an extensible sequence and structure informed deep learning framework for protein property prediction. *Scientific Reports*, 12, 2022.

Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.

Amir Shanehsazzadeh, David Belanger, and David Dohan. Is transfer learning necessary for protein landscape prediction? *arXiv preprint arXiv:2011.03443*, 2020.

A Supplementary Materials

A.1 Hyperparameters for pretrained models of different sizes

All models are trained for 2 weeks on 1-8 32GB V100 GPUs with dynamic batching. Table S1 summarizes the hyperparameters for CARP. Table S2 summarizes the hyperparameters for ESM.

Table S1: CARP model hyperparameters. Max tokens is the maximum number of tokens per GPU per batch during training.

Model	Parameters	Layers	d	d_{MLP}	Max tokens	GPUs
CARP-640M	643M	56	1280	1280	11000	128 × 32GB V100
CARP-76M	75.7M	32	1024	512	60000	16 × 32 GB V100
CARP-38M	37.9M	16	1024	512	40000	8 × 32 GB V100
CARP-24M	23.9M	16	256	128	400000	2 × 32 GB V100
CARP-600k	608k	16	128	64	600000	1 × 32 GB V100
CARP-40k	415k	16	32	16	600000	1 × 32 GB V100
CARP-4k	3670	16	8	4	600000	1 × 16 GB V100

Table S2: ESM model hyperparameters.

Parameters	Layers	Heads	d	d_{MLP}	GPUs
86.5M	12	12	768	3972	8 × 32 GB V100
44.0M	6	12	768	3072	8 × 32 GB V100
2.92M	6	8	192	768	2 × 32 GB V100
561k	4	6	96	384	1 × 32 GB V100
41.5k	4	4	24	96	1 × 32 GB V100
4890	2	4	4	16	1 × 32 GB V100

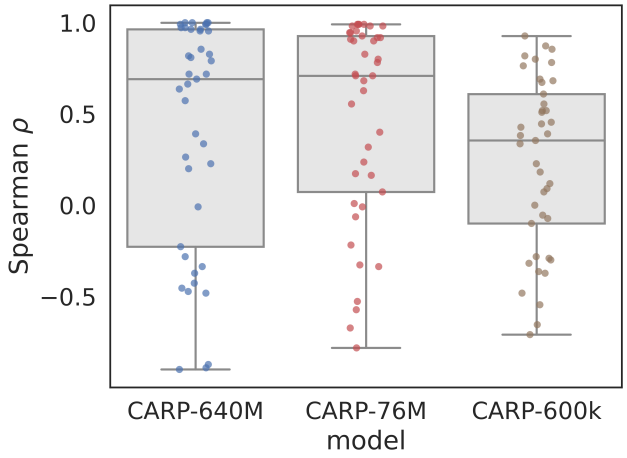
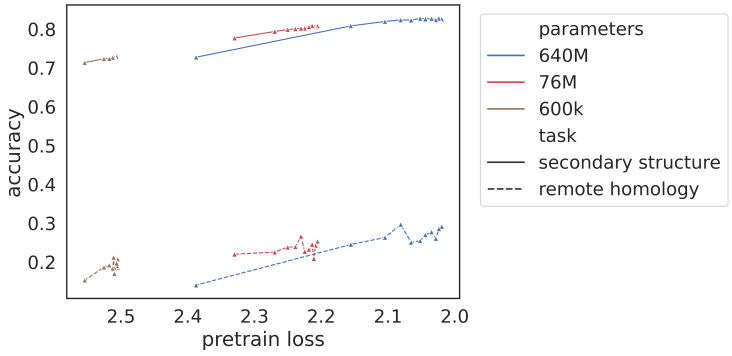
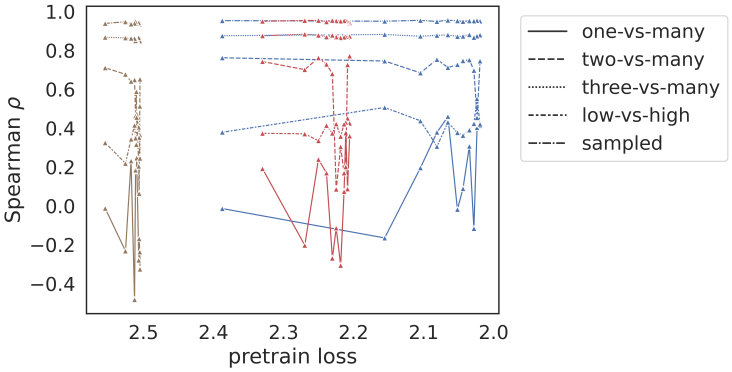


Figure S1: Spearman correlation between pretrained model checkpoint loss and zero-shot Spearman correlation across 41 deep mutational scanning datasets from DeepSequence.

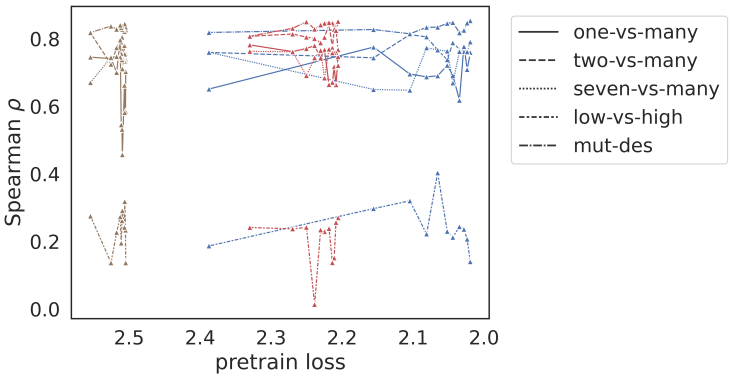
A.2 Pretrain performance vs downstream performance



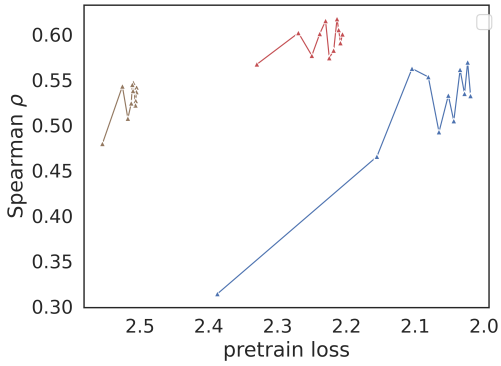
(a) Structure



(b) GB1



(c) AAV



(d) Meltome

Figure S2: Downstream performance vs pretrain loss for secondary structure, remote homology, and FLIP tasks.