Self-Supervised Contrastive Learning of Protein Representations By Mutual Information Maximization

Amy X. Lu [1 2] **Haoran Zhang** [1 2] **Marzyeh Ghassemi** [1 2] **Alan Moses** [1 3]

Abstract

Pretrained embedding representations of biological sequences which capture meaningful properties can alleviate many problems associated with supervised learning in biology. We apply the principle of mutual information maximization between local and global information as a self-supervised pretraining signal for protein embeddings. To do so, we divide protein sequences into fixed size fragments, and train an autoregressive model to distinguish between subsequent fragments from the same protein and fragments from random proteins. Our model, CPCProt, achieves comparable performance to state-of-the-art self-supervised models for protein sequence embeddings on various downstream tasks, but reduces the number of parameters down to 0.9% to 8.9% of benchmarked models. Further, we explore how downstream assessment protocols affect embedding evaluation, and the effect of contrastive learning hyperparameters on empirical performance. We hope that these results will inform the development of contrastive learning methods in protein biology and other modalities.

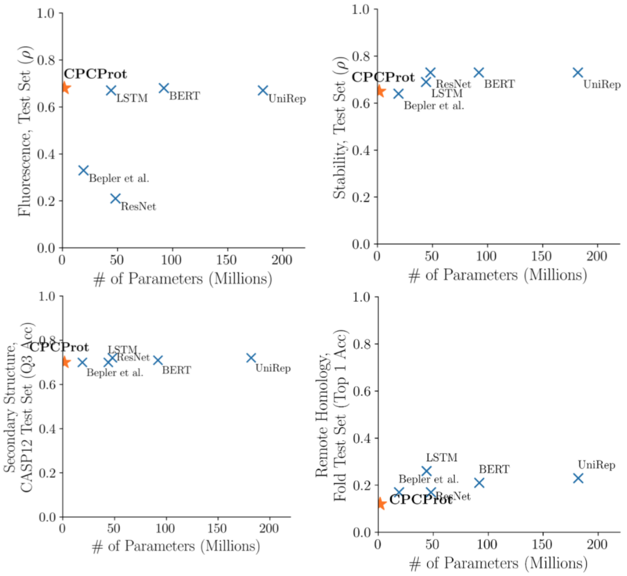

Figure 1. Downstream performance on protein-related tasks obtained by finetuning embeddings from pretrained models, plotted against the number of parameters in the pretrained model. Orange stars denote our model, and blue crosses denote methods benchmarked in Rao et al. [29]. ρ denotes Spearman's correlation for regression tasks.

1. Introduction

The cost and time associated with obtaining labels for *supervised* problems on proteins is a challenge, rendering self-supervised methods for obtaining labels an appealing solution.Though recent works have successfully demonstrated the ability to capture properties such as fluorescence, pairwise contact, phylogenetics, structure, and subcellular localization, these works mostly use methods designed for natural language processing (NLP) [41, 31, 32, 3, 16, 11, 4, 24, 12]. Presumably, using a more biologically-motivated proxy task will yield better insights and performance on biological data.

Biological sequences are inherently vehicles for information transmission. DNA transmits information across generations, which is then decoded into proteins by the central dogma. Across the "noisy channels" of heredity, DNA replication, transcription, and peptide signalling, genetic diversity offers useful redundancy which "error-corrects" against mutations which corrupt the functional and structural information conveyed. This analogy between Shannon information theory and biological proceses have been well-studied since the the 1970s [14, 20, 23, 2, 39], and successfully empirically applied in problems such as sequence logo visualization [35], transcription factor binding site discovery [36], structure prediction of protein loops [22], and evolutionary conservation of sequence features. [28, 27]. Viewing representation learning of proteins from the information theoretic lens is therefore arguably better rooted in

[1] Department of Computer Science, University of Toronto [2] Vector Institute for Artificial Intelligence [3] Department of Cell and Systems Biology, University of Toronto. Correspondence to: Alan Moses <alan.moses@utoronto.ca>.

biochemical realities than NLP-derived methods [3, 32, 29].

In this work, we present CPCProt, which maximize mutual information between context and local embeddings by minimizing a contrastive loss. On protein-related downstream benchmarks [29], CPCProt achieves comparable results, despite using 0.9% the number of pretraining parameters of the largest model [3] and 8.9% of the number of parameters of the smallest neural network model [6], as illustrated in Figure 1.

2. Methods

We describe CPCProt, which applies the InfoNCE loss introduced by the Contrastive Predictive Coding (CPC) method [26] to protein sequences. Pretrained model weights are available at `hershey.csb.utoronto.ca/CPCprot/weights/`, and code for our method is available at `http://github.com/amyxlu/CPCProt`.

2.1. Contrastive Predictive Coding and InfoNCE

The CPC method [26] introduces a lower-bound estimator for unnormalized mutual information. Define an encoder and autoregressor as g_{enc} and g_{ar}. Further, define x as an input protein sequence, z as the latent embedding produced by $g_{enc}(x)$, and c as the long-range protein sequence "context", as summarized by the autoregressor $g_{ar}(z)$. At a given position t (indexed for z), we estimate mutual information using the InfoNCE estimator $I'_{NCE}(z_{t+k}; c_t)$ for $k \in \{1, 2, \ldots, K\}$ by minimizing the loss:

$$\mathcal{L}_{t+k} = -\mathbb{E}\Big[\log \frac{\exp(f(z_{t+k}, c_t))}{\exp(f(z_{t+k}, c_t)) + \sum_{j=1}^{N-1} \exp(f(z'_j, c_t))}\Big]$$

In other words, in each batch of N samples, we have a single sample z_{t+k} drawn jointly with c_t from $p(z_{t+k}, c_t)$. Then, following the NCE method, we draw $N-1$ "fake" samples from the noise distribution $p(z')$ to create a set of $\{z'_j\}_{j=1}^{N-1}$. In practice, the expectation is taken over multiple batches.

This objective is a contrastive task, using a cross-entropy loss which encourages a critic, f, to correctly identify the single "real" sample of z_{t+k}. Minimizing this loss provides an unnormalized lower-bound estimate on the true MI, $I'_{NCE}(z_{t:(t+K)}; c_t)$ [26].

2.2. CPCProt: Applying InfoNCE to Protein Datasets

Each input x is divided into fixed-length patches, and each patch is encoded to output a single embedding for the patch, which are concatenated into the latent embedding z; that is, the length of z becomes $L_z = \lfloor \frac{\text{sequence\_length}}{\text{patch\_length}} \rfloor$. Here, a patch length of 11 is selected, such that it is long enough to capture local structural information and gives a reasonable L_z for the Pfam sequences we pretrain on. We start with some t_{min} to allow c_t to gain some context when calculating the loss, then calculate I'^{NCE}_{t+k} for every $t \in \{t_{\text{min}}, t_{\text{min}} + 1, ..., L_z - K\}$ and $k \in \{1, 2, ..., K\}$. A schematic detailing the method is illustrated in Figure 2.

The final loss minimized in each batch is the average of calculated $\mathcal{L}_{t+k}$ for all values of t and k:

$$\mathcal{L} = \frac{1}{L_z - K - t_{min}} \frac{1}{K} \sum_{t=t_{min}}^{L_z-K} \sum_{k=1}^{K} \mathcal{L}_{t+k} \tag{1}$$

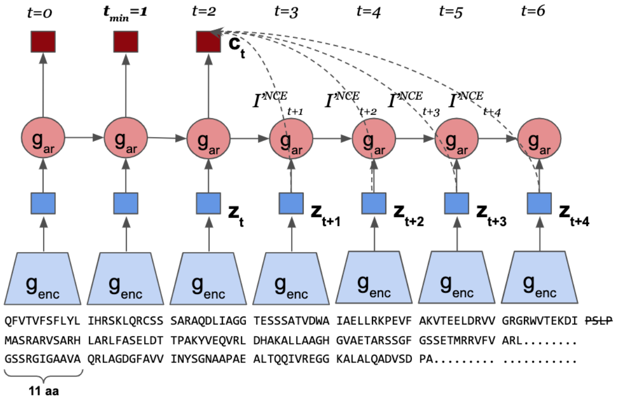
t=0
t_{min}=1
t=2
t=3
t=4
t=5
t=6
c_t
I'^{NCE}_{t+1}
I'^{NCE}_{t+2}
I'^{NCE}_{t+3}
I'^{NCE}_{t+4}
g_{ar}
z_t
z_{t+1}
z_{t+2}
z_{t+3}
z_{t+4}
g_{enc}
QFVTVFSFLYL IHRSKLQRCSS SARAQDLIAGG TESSSATVDWA IAELLRKPEVF AKVTEELDRVV GRGRWVTEKDI ~~PSLP~~
MASRARVSARH LARLFASELDT TPAKYVEQVRL DHAKALLAAGH GVAETARSSGF GSSETMRRVFV ARL........
GSSRGIGAAVA QRLAGDGFAVV INYSGNAAPAE ALTQQIVREGG KALALQADVSD PA......... ....
11 aa

Figure 2. Input protein sequences are divided into "patches" of 11 amino acids. Each patch is encoded into a single vector though g_{enc}, and all encodings are concatenated to form z. g_{ar} is an autoregressor that aggregates local information, and produce c, a context vector which summarizes the global context. Amino acid sequences are zero-padded to the longest sequence in the batch, with remaining amino acids not long enough for a patch discarded. For a given batch, the loss is the average of the InfoNCE estimate I'_{t+k} for all $t \in \{t_{\text{min}}, t_{\text{min}}+1, ..., L_z - K\}$ and $k \in \{1, 2, ..., K\}$. In this example batch, $t_{\text{min}} = 1$, $L_z = 6$, and $K = 4$.

Architecture The CPCProt encoder consists of embedding layer to 32 hidden dimensions, followed by 64 filters of length 4, 64 filters of length 6, and 512 filters of length 3. Filter lengths are designed such that the output of an 11 amino acid patch has an output length of one. A single-layer GRU is used as the autoregressor. All autoregressors use the same number of hidden dimensions as encoder output.

Additional Pretraining Details Models are pretrained on protein domain sequences from the Pfam database [10] using the same data splits as used for downstream benchmarks [29]. We use $t_{\text{min}} = 1$ and choose $K = 4$ (that is, 44 amino acids away). Sequences are zero-padded up to the longest sequence in that batch, and truncated to a maximum of 550 amino acids. We use a parameterless dot product critic for f and draw "fake" samples from $p(z)$ and $p(c)$ using other z_{t+k} and c_t from other samples in the same batch [7]. A batch size of 64 is used, trained for 19 epochs with a constant learning rate of 1e-4 using the Adam optimizer.

In addition, we report results using a large encoder and a LSTM autoregressor, with architectural and pretraining details in Appendix A.

3. Data and Downstream Evaluation

Evaluating Embeddings for Hyperparameter Selection To avoid overfitting to benchmarks [30], we examine three evaluations for pretraining hyperparameter selection: (1) Validation set performance on benchmarked downstream tasks; (2) contrastive accuracy during pretraining; and (3) Pfam family prediction using a 1-nearest-neighbor (1NN) classifier on the pretraining validation data. See Appendix B for further details.

Models for Downstream Evaluation For consistency with benchmarks, we use the same default downstream architectures as provided by the authors [29]. Following previous work in self-supervised learning, we also assess linear separability of embeddings using a linear model [26, 17, 15, 5, 37, 38]. Note that as compared to the neural network finetuning head evaluation, we use static embeddings extracted from the model without end-to-end optimization. In addition, we evaluate separation in the latent space using a kNN model with a grid search over the number of neighbours $k \in \{1, 5, 10\}$.

Downstream Evaluation Tasks and Data We assess our models on four TAPE downstream benchmark tasks [29]: remote homology, secondary structure, fluorescence, and stability. See Appendix C for details.

4. Results

CPCProt Performs Comparably with Baselines Using Fewer Parameters CPCProt achieves comparable results as baselines on most tasks; however, we use only 0.9% of the number of embedding parameters of the largest model [3] and 8.9% of the number of embedding parameters of the smallest neural network [6] (Tables 1-5; Figure 1).

For the fluorescence task, CPCProt achieves higher ρ and lower MSE than other models for both a neural network finetuning head and most linear regression and kNN evaluations (Tables 1, 5). For the secondary structure and stability tasks, CPCProt achieves comparable performance with other neural network models, with a fraction of the number of parameters (Tables 1, 3, 4).

Downstream Assessment is Inconsistent Using Different Models For the purpose of using downstream tasks primarily as a means to evaluate the quality of embeddings, changing the downstream model used sometime result in preferring different models (Tables 1, 2, 3, 4, 5), as does using a different performance metric (i.e. MSE versus Spearman's ρ for regression tasks) (Tables 4, 5). For example, though CPCProt appear to generalize poorly to the Fold test set for the remote homology task relative to baselines using a neural network classifier head (Table 1), it in fact outperforms both BERT and UniRep when using a kNN classifier (Table 2). For fluorescence and stability, switching to a simple linear or kNN model better differentiate performances of UniRep, BERT, and CPCProt variants (Tables 4, 5). Performance improves when using a finetuned neural network head for most embeddings, but for secondary structure, finetuning with a higher capacity MLP actually *decreases* Q3 accuracy.

5. Discussion

Relationship Between Pretrained Model Size and Downstream Performance Table 1 and Figure 1 show that there is no clear connection between increasing the number of parameters (in the pretrained model only) and downstream performance, contrary to the philosophy behind 567 million parameter NLP-inspired models for protein representations [12]. This is true even for variants of CPCProt, which were trained using the same self-supervised objective.

The finding that CPCProt achieves comparable results with less parameters may be a reflection of the overparameterization of existing protein embedding models in the field, or of a unique benefit conferred by the contrastive training. In any case, these results show that simply porting large models from NLP to proteins is not an efficient use of computational resources, and we encourage the community to further explore this relationship. As compared to currently-available protein embedding models, we note the suitability of CPCProt for downstream use-cases where model size is a key concern.

Difficulties in Quantitatively Assessing Protein Embeddings As explored in Section 4, it is difficult to quantitatively assess embedding performance, as downstream performance differs by downstream model and performance metrics (i.e. MSE vs ρ). Moreover, it is difficult to attribute quantitative performance on downstream tasks to the information captured by the embedding, or to the supervised finetuning procedures. This is complicated by the inconsistency in whether if encoder weights should be frozen during training: while some works in contrastive learning freeze the encoder during finetuning [40], large NLP embedding models such as BERT typically update parameters end-to-end [9], as do protein models inspired by these NLP models [29].

	# of Embedding Parameters	Remote Homology			Secondary Structure			Stability	Fluorescence
		Fold	Superfamily	Family	CB513	CASP12	TS115		
Unirep	182M	0.23	0.38	0.87	0.73	**0.72**	0.77	**0.73**	0.67
BERT	92M	0.21	0.34	0.88	0.73	0.71	0.77	**0.73**	**0.68**
ResNet	48M	0.17	0.31	0.77	**0.75**	**0.72**	**0.78**	**0.73**	0.21
LSTM	44M	**0.26**	**0.43**	**0.92**	**0.75**	0.70	**0.78**	0.69	0.67
Bepler et al.	19M	0.17	0.20	0.79	0.73	0.70	0.76	0.64	0.33
One Hot	0	0.09	0.08	0.39	0.69	0.68	0.72	0.19	0.14
CPCProt	1.7M	0.12	0.12	0.48	0.69	0.70	0.73	0.65	**0.68**
CPCProt$_{\text{GRU\_large}}$	8.4M	0.13	0.14	0.52	0.70	0.70	0.73	0.65	**0.68**
CPCProt$_{\text{LSTM}}$	71M	0.11	0.11	0.47	0.68	0.66	0.70	0.68	**0.68**

Table 1. Embedding performance by downstream task using the default neural network finetuning head, compared against Tasks Assessing Protein Embeddings (TAPE) benchmarks [29].

	Remote Homology					
	Fold		Superfamily		Family	
	LR	kNN	LR	kNN	LR	kNN
UniRep	0.08	0.06	0.18	0.11	0.48	0.38
BERT	**0.20**	0.11	**0.30**	**0.24**	**0.76**	**0.74**
CPCProt	0.14	**0.12**	0.13	0.10	0.50	0.51
CPCProt$_{\text{GRU\_large}}$	0.13	**0.12**	0.14	0.10	0.50	0.55
CPCProt$_{\text{LSTM}}$	0.14	0.11	0.15	0.12	0.52	0.55

Table 2. Downstream evaluation using logistic regression and kNN k-nearest-neighbours models (Top-1 accuracy).

	Secondary Structure		
	CB513	CASP12	TS115
	LR	LR	LR
UniRep	0.66	0.80	0.70
BERT	**0.72**	**0.82**	**0.77**
CPCProt	0.61	0.80	0.68
CPCProt$_{\text{GRU\_large}}$	0.62	0.80	0.69
CPCProt$_{\text{LSTM}}$	0.62	0.80	0.69

Table 3. Downstream evaluation using logistic regression models. Top-3 (Q3) accuracy is reported.

	Stability			
	LR		kNN	
	MSE	ρ	MSE	ρ
UniRep	**0.21**	**0.62**	0.24	**0.57**
BERT	0.36	0.39	0.23	0.49
CPCProt	0.34	0.55	**0.18**	0.51
CPCProt$_{\text{GRU\_large}}$	0.31	**0.62**	**0.18**	0.52
CPCProt$_{\text{LSTM}}$	0.22	**0.62**	0.19	0.54

Table 4. Downstream evaluation using linear regression and kNN models for stability (MSE and Spearman's ρ).

	Fluorescence			
	LR		kNN	
	MSE	ρ	MSE	ρ
UniRep	1.32	0.55	**1.66**	0.37
BERT	1.15	0.52	1.75	0.46
CPCProt	1.13	0.54	1.82	0.49
CPCProt$_{\text{GRU\_large}}$	**0.81**	0.63	1.84	0.50
CPCProt$_{\text{LSTM}}$	0.85	**0.67**	1.80	**0.51**

Table 5. Downstream evaluation using linear regression and kNN models for fluorescence (MSE and Spearman's ρ).

We hope to highlight that downstream benchmarks should not definitively define utility of an embedding. Under consistent protocols, they may be good proxies to examine specific desiderata regarding global and local information or out-of-distribution generalization. Given the diversity of biological use cases, embedding evaluations should be made on a case-by-case basis.

6. Conclusion

In this work, we introduce CPCProt, which achieves comparable downstream performance as existing protein embedding models, at a fraction of the number of parameters. We further compare the effects of using different pretraining evaluation metrics and downstream models for evaluating embeddings on protein-related tasks, and find that there is poor consistency in how models compare against one another, illustrating the difficulty in defining the utility of an embedding for biological use cases. We hope that this work can inform the development of other embedding models for biological sequences.

Acknowledgements

Amy X. Lu was funded by the NSERC Canada Graduate Scholarships-Master's award. Alan M. Moses holds a Tier II Canada Research Chair. Maryzeh Ghassemi is funded in part by Microsoft Research, a CIFAR AI Chair at the Vector Institute, a Canada Research Council Chair, and an NSERC Discovery Grant. Computing resources for this work was provided by the Vector Institute. The authors thank Nick Bhattacharya, Alex X. Lu, Adam Riesselman, Denny Wu, and Kevin Yang for insightful input.

References

[1] Abriata, L. A., Tamò, G. E., Monastyrskyy, B., Kryshtafovych, A., and Dal Peraro, M. Assessment of hard target modeling in casp12 reveals an emerging role of alignment-based contact prediction methods. *Proteins: Structure, Function, and Bioinformatics*, 86: 97–112, 2018.

[2] Adami, C. Information theory in molecular biology. *Physics of Life Reviews*, 1(1):3–22, 2004.

[3] Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. Unified rational protein engineering with sequence-only deep representation learning. *bioRxiv*, pp. 589333, 2019.

[4] Armenteros, J. J. A., Johansen, A. R., Winther, O., and Nielsen, H. Language modelling for biological sequences–curated datasets and baselines. 2019.

[5] Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pp. 15509–15519, 2019.

[6] Bepler, T. and Berger, B. Learning protein sequence embeddings using information from structure. *arXiv preprint arXiv:1902.08661*, 2019.

[7] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[8] Cuff, J. A. and Barton, G. J. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins: Structure, Function, and Bioinformatics*, 34(4):508–519, 1999.

[9] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[10] El-Gebali, S., Mistry, J., Bateman, A., Eddy, S. R., Luciani, A., Potter, S. C., Qureshi, M., Richardson, L. J., Salazar, G. A., Smart, A., Sonnhammer, E. L. L., Hirsh, L., Paladin, L., Piovesan, D., Tosatto, S. C. E., and Finn, R. D. The Pfam protein families database in 2019. *Nucleic Acids Research*, 47(D1):D427–D432, 2019. ISSN 0305-1048. doi: 10.1093/nar/gky995. URL https://academic.oup.com/nar/article/47/D1/D427/5144153.

[11] Elnaggar, A., Heinzinger, M., Dallago, C., and Rost, B. End-to-end multitask learning, from protein language to protein features without alignments. *bioRxiv*, pp. 864405, 2019.

[12] Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Bhowmik, D., et al. Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing. *arXiv preprint arXiv:2007.06225*, 2020.

[13] Fox, N. K., Brenner, S. E., and Chandonia, J.-M. Scope: Structural classification of proteins—extended, integrating scop and astral data and classification of new structures. *Nucleic acids research*, 42(D1):D304–D309, 2013.

[14] Gatlin, L. L. et al. *Information theory and the living system*. Columbia University Press, 1972.

[15] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[16] Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., and Rost, B. Modeling the language of life-deep learning protein sequences. *bioRxiv*, pp. 614313, 2019.

[17] Hénaff, O. J., Razavi, A., Doersch, C., Eslami, S., and Oord, A. v. d. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

[18] Hou, J., Adhikari, B., and Cheng, J. Deepsf: deep convolutional neural network for mapping protein sequences to folds. *Bioinformatics*, 34(8):1295–1303, 2018.

[19] Kabsch, W. and Sander, C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers: Original Research on Biomolecules*, 22(12):2577–2637, 1983.

[20] Kimura, M. Natural selection as the process of accumulating genetic information in adaptive evolution. *Genetics Research*, 2(1):127–140, 1961.

[21] Klausen, M. S., Jespersen, M. C., Nielsen, H., Jensen, K. K., Jurtz, V. I., Soenderby, C. K., Sommer, M. O. A., Winther, O., Nielsen, M., Petersen, B., et al. Netsurfp-2.0: Improved prediction of protein structural features by integrated deep learning. *Proteins: Structure, Function, and Bioinformatics*, 2019.

[22] Korber, B. T., Farber, R. M., Wolpert, D. H., and Lapedes, A. S. Covariation of mutations in the v3 loop of human immunodeficiency virus type 1 envelope protein: an information theoretic analysis. *Proceedings of the National Academy of Sciences*, 90(15):7176–7180, 1993.

[23] MacKay, D. J. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[24] Madani, A., McCann, B., Naik, N., Keskar, N. S., Anand, N., Eguchi, R. R., Huang, P.-S., and Socher, R. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

[25] Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. Critical assessment of methods of protein structure prediction (CASP)-Round XII. *Proteins: Structure, Function, and Bioinformatics*, 86:7–15, 2018. ISSN 08873585. doi: 10.1002/prot.25415. URL http://doi.wiley.com/10.1002/prot.25415.

[26] Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[27] Pritišanac, I., Vernon, R. M., Moses, A. M., and Forman Kay, J. D. Entropy and information within intrinsically disordered protein regions. *Entropy*, 21(7):662, 2019.

[28] Rao, G. S., Hamid, Z., and Rao, J. S. The information content of dna and evolution. *Journal of theoretical biology*, 81(4):803–807, 1979.

[29] Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, P., Canny, J., Abbeel, P., and Song, Y. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, pp. 9686–9698, 2019.

[30] Recht, B., Roelofs, R., Schmidt, L., and Shankar, V. Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811*, 2019.

[31] Riesselman, A. J., Shin, J.-E., Kollasch, A. W., McMahon, C., Simon, E., Sander, C., Manglik, A., Kruse, A. C., and Marks, D. S. Accelerating protein design using autoregressive generative models. *bioRxiv*, pp. 757252, 2019.

[32] Rives, A., Goyal, S., Meier, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, pp. 622803, 2019.

[33] Rocklin, G. J., Chidyausiku, T. M., Goreshnik, I., Ford, A., Houliston, S., Lemak, A., Carter, L., Ravichandran, R., Mulligan, V. K., Chevalier, A., et al. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357(6347):168–175, 2017.

[34] Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., et al. Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397, 2016.

[35] Schneider, T. D. and Stephens, R. M. Sequence logos: a new way to display consensus sequences. *Nucleic acids research*, 18(20):6097–6100, 1990.

[36] Stormo, G. D. Dna binding sites: representation and discovery. *Bioinformatics*, 16(1):16–23, 2000.

[37] Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[38] Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning. *arXiv preprint arXiv:2005.10243*, 2020.

[39] Vinga, S. Information theory applications for biological sequence analysis. *Briefings in bioinformatics*, 15 (3):376–389, 2014.

[40] Wang, T. and Isola, P. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *arXiv preprint arXiv:2005.10242*, 2020.

[41] Yang, K. K., Wu, Z., Bedbrook, C. N., and Arnold, F. H. Learned protein embeddings for machine learning. *Bioinformatics*, 34(15):2642–2648, 2018.

[42] Yang, Y., Gao, J., Wang, J., Heffernan, R., Hanson, J., Paliwal, K., and Zhou, Y. Sixty-five years of the long march in protein secondary structure prediction: the final stretch? *Briefings in bioinformatics*, 19(3): 482–494, 2018.

A. Architectural Variants

In addition to the main CPCProt model we present in Figure 1, we also report results using a larger encoder. Both $\text{CPCProt}_{\text{GRU\_large}}$ and $\text{CPCProt}_{\text{LSTM}}$ uses an embedding layer to 64 hidden dimensions, followed by 128 filters of length 4, 256 filters of length 4, and 512 filters of length 3; in the final layer, $\text{CPCProt}_{\text{GRU\_large}}$ uses 1024 filters of length 3, whereas $\text{CPCProt}_{\text{LSTM}}$ uses 2048 filters. For $\text{CPCProt}_{\text{LSTM}}$, a two-layer LSTM is used instead of a GRU. To avoid information leakage about later sequence locations in the context vector, we only use uni-directional autoregressors.

B. Evaluation for Pretraining Hyperparameter Selection

Though in principle, the contrastive accuracy on heldout pretraining data is sufficient for hyperparameter selection, we were concerned that the contrastive task is relatively local, and may fail to assess how well embeddings have captured the global context. We also wanted to avoid overfitting to downstream benchmarks. For the contrastive task (i.e. the self-supervised pretraining task), we keep the ratio of negative-to-positive samples consistent across models, and use a batch size of 512 for all models for this validation.

The 1NN classification task is a direct measure of the ability for embeddings to cleanly separate Pfam domains in the latent space, and requires no parameter tuning or additional labels for evaluation. For this task, the dataset consists of sequences from the 50 Pfam families with the most sequences in the pretraining validation dataset, subsampled to 120 sequences per family for class balance. 70% of this embeddings is used to populate the 1NN classifier, and 30% of the sequences are used at the classification phase. A t-SNE of CPCProt embeddings colored by the 50 families is shown in Figure 3.

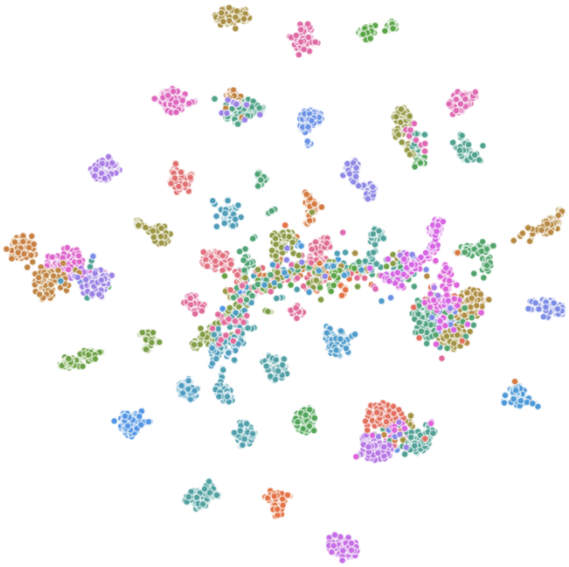

Figure 3. CPCProt embedding t-SNE of the 50 largest Pfam families in the validation dataset, using the final position of the context vector. Note that while colours denote different families, proximity in the continuous color space do not correspond to any intrinsic similarities between families.

Our main CPCProt model (1.7M parameters), from which all analyses are derived, was selected as it had the best overall performance on the downstream validation set. We also report the model with the best contrastive task accuracy ($\text{CPCProt}_{\text{LSTM}}$) and with the best 1NN task accuracy ($\text{CPCProt}_{\text{GRU\_large}}$).

C. Data Description for Downstream Tasks

Remote Homology Remote sequence homologs share conserved structural folds but have low sequence similarity. The task is a multi-class classification problem, consisting of 1195 classes, each corresponding to a structural fold. Since global context from across the Pfam domain is important, we use the final position of the autoregressor output, c.

Data from the SCOP 1.75 database [13] is used. Each fold can be sub-categorized into superfamilies, and each superfamily

	Pretraining Validation Dataset		**Downstream Tasks Validation Dataset**			
	Contrastive Accuracy	**1NN**	**Fluorescence**	**Stability**	**Remote Homology**	**Secondary Structure**
CPCProt	0.04	0.89	**0.80**	**0.68**	0.11	**0.71**
CPCProt$_{\text{LSTM}}$	**0.09**	0.91	0.79	0.68	**0.12**	0.66
CPCProt$_{\text{GRU\_large}}$	0.05	**0.93**	0.72	0.67	0.10	0.69

Table 6. Results on evaluation metrics for selecting hyperparameters and architectures. The contrastive accuracy is the NCE task, using a batch size of 1024, such that a random model achieves an expected accuracy of 0.00098. Our CPCProt$_{\text{LSTM}}$ variant achieves the highest contrastive accuracy of all evaluated models, while the CPCProt$_{\text{GRU\_large}}$ variant achieves the highest accuracy on the 1NN Pfam family prediction task. We report models selected by metrics which do not depend on downstream tasks to avoid benchmark overfitting, and to examine the effect of increasing the number of parameters on desired properties of embeddings. For reference, UniRep achieves 95% accuracy on our 1NN Pfam family prediction task and dataset.

sub-categorized into families. The training, validation, and test set splits are curated in Hou et al. [18]; test sets examines three levels of distribution shift from the training dataset. In the "Family" test set, proteins in the same fold, superfamily, and family exists in both the training and testing datasets (i.e. no distribution shift). The "Superfamily" test set holds out certain families within superfamilies, but sequences with overlap with training dataset at the superfamily level. Finally, the "Fold" test set also holds out certain superfamilies within folds. Note that severe class imbalance exists for this task, as 433 folds in the training dataset only contains one sample.

For evaluation using a neural network head, the classification architecture is a multi-layer perceptron (MLP) with one hidden layer of 512 units. with ReLU activation and weight normalization. Note that results in benchmarked models also train a simple dense layer to obtain an attention vector before calculating an attention-weighted mean.

Secondary Structure Secondary structure is a sequence-to-sequence task evaluating the embeddings' ability to capture local information [29]. We report three-class accuracy (Q3), following the DSSP labeling system [19]. Each input amino acid is mapped to one of three labels ("helix", "strand", or "other"), and accuracy is the percentage of correctly-labelled positions. To obtain the embedding, we use a sliding input window to obtain z with the same length as the input sequence, and then use c as the embedding to incorporate global context.

Classification results are presented on three datasets: (1) TS115, consisting of 115 protein sequences [42]; (2) CB513, consisting of 513 protein regions from 434 proteins [8]; and (3) free-modelling targets from the 2016 CASP12 competition, consisting of 21 protein sequences [1, 25]. For training these supervised classifiers, the same validation and filtered training datasets as NetSurf-2.0 is used, where sequences with greater than 25% sequence similarity as the three test set sequences were removed from the training set 500 sequences randomly heldout for validation, leaving 10,337 sequences for training [21].

For evaluation using a neural network head, the classification architecture in `tape-proteins` is a convolutional architecture with 512 filters of size 5 and 3 in layers one and two, respectively. The original benchmarks use a higher capacity NetSurfP model [21], with two convolutional layers followed by two bidirectional LSTM layers and a linear output layer.

Fluorescence The fluorescence task is a protein engineering task which evaluates how fine-trained local genotypic changes can be captured to predict phenotypic expression, as measured by native fluorescence. The regression task is to predict the log-intensity of a mutant GFP sequence. Since this task is more sensitive to local than global information, we apply a mean-pool along the sequence dimension of the encoder output, z.

The data is from a Deep Mutational Scan (DMS) experiment from Sarkisyan et al. [34], which measures fluorescence from derivative genotypes of the green fluorescent protein avGFP. Data splits are curated in Rao et al. [29]. Training and validation data are in a Hamming distance 3 neighborhood from the original protein, while the test data exhibits larger distribution shift and is from the Hamming distance 4-15 neighborhood.

For evaluation using a neural network head, `tape-proteins` uses the same MLP architecture as described in the remote homology task. The original benchmarks in Rao et al. [29] compute an trainable attention-weighted mean prior to classification.

Stability Stability is a protein engineering task which measures the most extreme concentration for which a protein can maintain its structure. This is a regression task to predict a stability score of proteins generated by *de novo* design. Since this task is also sensitive to fine-grained local effects, we use the mean along the encoder output z as a pooled embedding.

The data is from Rocklin et al. [33], which measures the stability of proteins generated by parallel DNA synthesis, consisting of sequences from four protein topologies: $\alpha\alpha\alpha$, $\beta\alpha\beta\beta$, $\alpha\beta\beta\alpha$, $\beta\beta\alpha\beta\beta$. The stability score is the difference between the measured EC_{50} of the actual protein and its predicted EC_{50} in its unfolded state. Here, EC_{50} is the protease concentration at which 50% of cells pass the characterization threshold; note that it is measured on a $\log_{10}$ scale. Data splits are curated in Rao et al. [29], such that the test set consists of seventeen 1-Hamming distance neighbourhoods from the training and validation datasets. A visualization of this test split is shown in Figure 4.

For evaluation using a neural network head, as with remote homology and fluorescence, we use the provided MLP architecture, while the original benchmarks compute an trainable attention-weighted mean prior to classification.

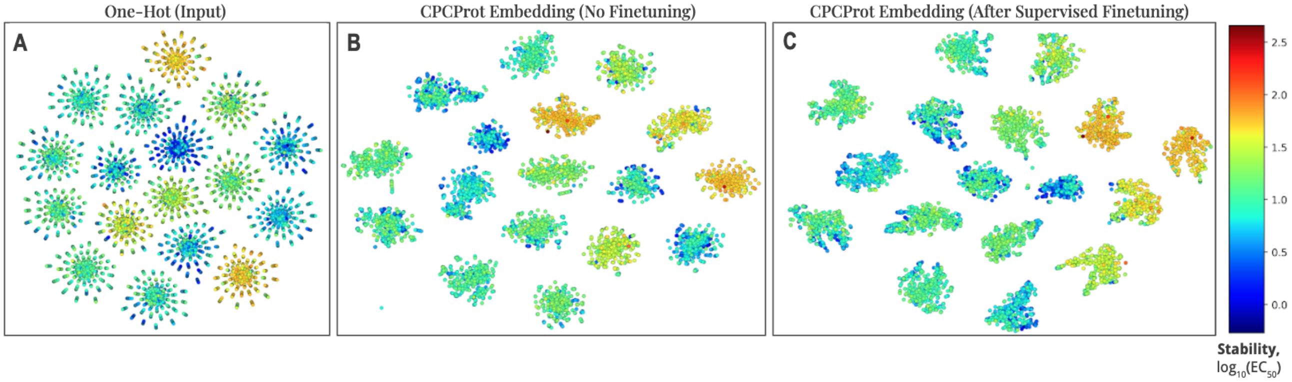

Figure 4. t-SNE visualization of proteins which are all 1-Hamming distance away from one of seventeen candidate proteins. Colors denote stability measurement on a $\log_{10}$ scale. The data corresponds to test set curated in the TAPE benchmarks [29] for the stability dataset from Rocklin et al. [33].