# Training Compute-Optimal Protein Language Models

Xingyi Cheng [1] Bo Chen [1 2] Pan Li [1] Jing Gong [1] Jie Tang [2] Le Song [1 3]

## Abstract

We explore the optimal training of protein language models, an area crucial in biological research where guidance is limited. Most models are trained with extensive compute resources, emphasizing model size increases over efficient compute usage. Our research uses a large dataset of 939 million protein sequences, training over 300 models ranging from 3.5 million to 10.7 billion parameters on 5 to 200 billion tokens to examine the relationships between model sizes, token numbers, and objectives. Initial findings show diminishing returns for Causal Language Models (CLM) and overfitting tendencies in Masked Language Models (MLM) when using the Uniref database. To combat this, we incorporated metagenomic sequences to diversify the training set and mitigate plateauing and overfitting. We derived scaling laws for CLM and MLM on Transformers, tailored to protein sequence data characteristics. To validate these scaling laws, we compared large-scale ESM-2 and PROGEN2 models in downstream tasks, including protein generation and structure- and function-related evaluations, within comparable pre-training compute budgets.

## 1. Introduction

Scaling attention-based transformers, especially in Natural Language Processing (NLP) (Brown et al., 2020b; Touvron et al., 2023a; Rae et al., 2021; Taylor et al., 2022) and Computer Vision (CV) (Zhai et al., 2022; Riquelme et al., 2021; Dehghani et al., 2023), enhances model performance. Similarly, large Transformer-based Protein Language Models (PLMs) like the PROGEN (Madani et al., 2020; Nijkamp et al., 2023) and ESM families (Rives et al., 2021; Lin et al., 2023), and xTrimoPGLM (Chen et al., 2024) have significantly advanced performance in complex tasks (Li et al., 2024; Elnaggar et al., 2023). These models employ BERT-like Masked Language Model (MLM) (Devlin et al., 2018) and GPT-like Causal Language Model (CLM) (Brown et al., 2020a) objectives. Bi-directionally encoded MLM excels in sample efficiency and fine-tuning for downstream tasks (Lin et al., 2022; Chen et al., 2024). In contrast, uni-directional CLM is better suited for generating coherent sequences (Dauparas et al., 2022; Nijkamp et al., 2023; Qiu et al., 2024). However, allocating compute budgets optimally for training PLMs is relatively underexplored, with *most efforts focusing on scaling model parameters based on a fixed set of training tokens to achieve performance improvements*. A key insight (Kaplan et al., 2020; Hoffmann et al., 2022; Tay et al., 2021) is that large models should not be trained to their lowest possible loss to optimize computing; instead, models and data should be scaled proportionally based on available compute budgets. These scaling laws are broadly found in natural language models (Kaplan et al., 2020; Hoffmann et al., 2022). But their applicability has not been validated within biological datasets, such as the primary structures of proteins, which are composed of amino acid sequences forming protein chains. Thus, we consider such data as a distinct modality and ask the question: *What are the scaling behaviors for MLM and CLM in protein language modeling?*

Our study revisits datasets, objectives, and parameters to optimize training schemes for PLMs within set compute budgets. Key findings include: (1) We collected a dataset of 194 billion unique tokens across 939M sequences to mitigate overfitting in protein language modeling. (2) Training data scales sublinearly with model size; MLM and CLM follow distinct power-laws. For example, a 10× compute increase leads to a 6× MLM model size increase and 70% more data, compared to a 4× increase in CLM model size and 3× in training tokens. (3) Applying our scaling strategies, we optimized the PROGEN2-xlarge and ESM-2 (3B) setups, training two models with 7.2B and 10.7B parameters, respectively, improving performance in various downstream tasks.

## 2. Scaling up data

We explores training PLMs over multiple epochs with the introduction of the UniMeta200B dataset, enhancing training efficiency for protein language models.

### 2.1. A Data-hungry Observation

Using the UniParc database with 250 million protein sequences, research on ESM demonstrates that diverse datasets UR50/S and UR50/D, with respective unique se-

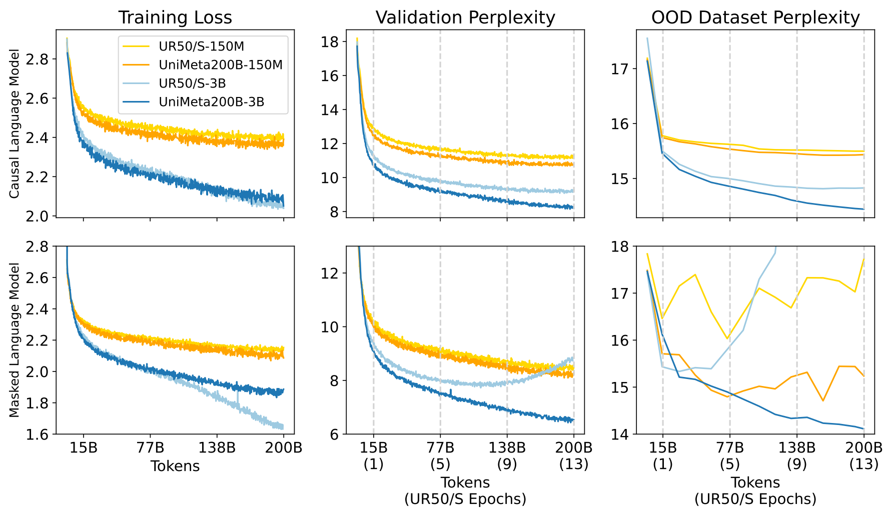

*Figure 1.* **Learning curves for UR50/S and UniMeta200B.** Training loss and validation PPL, OOD test PPL were tracked for 200 billion tokens across models with 150M and 3B parameters. Scaling from 150M to 3B parameters showed diminishing returns on CLM and a tendency for MLM to overfit. Three repetition methods evaluated on the 3B MLM models all exhibited overfitting (see Appendix D).

*Table 1.* **The Pre-training data**, aggregates various public sources and specifies sampling proportions for a single epoch of training on 194 billion unique amino acids.

| Datasets | Prot. Seq. | Tokens (AAs) | Samp. Prop. |
|---|---|---|---|
| Uniref50/S | 54M | 15.2B | 8.5% |
| Uniref90/50 | 102M | 37.8B | 19.5% |
| ColabFoldDB$_c$ | 208M | 37.7B | 19.5% |
| ColabFoldDB$_m$ | 575M | 103B | 52.5% |
| Total | 939M | 194B | - |

quence counts of 45M and 65M, surpass Uniref100 in MLM perplexity on a ~670M parameter model (Rives et al., 2021). These datasets comprise ~15B and ~20B unique amino acid tokens. ESM-2 family models, ranging from 150M to 15B parameters, are extensively trained with nearly 1 trillion tokens over 45 epochs on the UR50/D dataset. In contrast, contemporary LLMs typically train for only a few epochs (Brown et al., 2020a; Komatsuzaki, 2019), highlighting the challenges of scaling models due to data repetition and token limitations (Raffel et al., 2020; Hernandez et al., 2022).

### 2.2. Expanding Diversified Metagenomic Data

Given that the Uniref90 dataset has proven most effective for pre-training across various Uniref clustering levels per ESM-1v (Meier et al., 2021), we incorporated Uniref90/50 (Before 2022-12), which includes incremental data relative to Uniref50/S representatives. ColabFoldDB$_c$ and ColabFoldDB$_m$ play dominant roles within the dataset, corresponding to cluster representatives and members, respectively. To ensure uniformity during training, we allocate weights within each batch to allow each amino acid token to be evenly processed through the model. This dataset, termed UniMeta200B, contains 939 million unique protein sequences and 194 billion amino acids, which is an order of magnitude larger than UR50/D. We observed significant improvements in the OOD test set and a consistent learning curve on the IID validation subset extracted from the training set (Figure 1) [1].

## 3. Scaling laws for CLM and MLM

*Table 2.* Coefficient of Equation 1.

| Parameter | $\alpha$ | $\beta$ | $A$ | $B$ |
|---|---|---|---|---|
| CLM | 0.578 | 0.422 | $1.26 \times 10^{-3}$ | $1.23 \times 10^{2}$ |
| MLM | 0.776 | 0.230 | $6.19 \times 10^{-8}$ | $2.02 \times 10^{6}$ |

We fit our models in the form of a fundamental power-law based on the existing work (Kaplan et al., 2020; Hoffmann et al., 2022) in the field of LLMs. Specifically, given a fixed FLOPs formula of $C = 6 \times N \times D$, where $N$ represents the number of forward-activated non-embedding parameters, and $D$ is the number of training tokens, how should one navigate the trade-off between model size and the number of training tokens? The model parameters $N$ and data size $D$ can be directly fit with a simple power-law:

$$N(C) = A \times C^{\alpha}, \quad D(C) = B \times C^{\beta} \tag{1}$$

[1]Appendix G compare the training performed separately on two datasets, and we find that the ColabFoldDB does not affect downstream results.

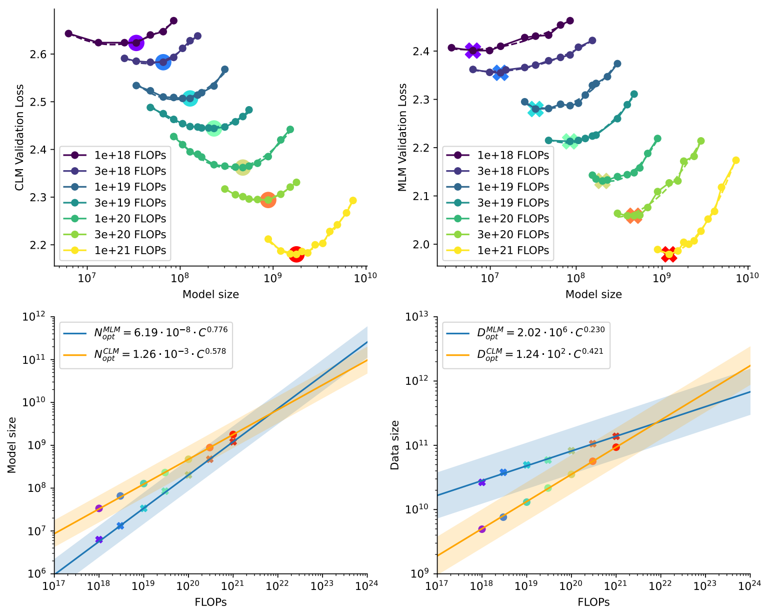

*Figure 2.* **IsoFLOPs curves and parametric fit for CLM and MLM.** We selected training tokens to ensure a uniform final FLOP count for different model sizes. The lowest loss of each curve revealed an optimal model size for a FLOP budget (**above**). We use these rainbow points at the valley to plot the efficient frontier for estimating the optimal model size and training tokens for scaling models (**below**). The interval range was estimated by model points with similar loss.

We employed the IsoFLOPs profiling approach (Hoffmann et al., 2022; Bi et al., 2024), setting 7 distinct training FLOP counts ranging from $1 \times 10^{18}$ to $1 \times 10^{21}$. For each FLOP count, we selected models from a pool of candidates (see Appendix Q). Models were excluded if the estimated data size $(C/(6 * N))$ resulted in more than 200B tokens or if the training steps were fewer than 20K. Ultimately, approximately 260 models were used for fitting. We considered the final validation loss for each model to ensure that every model completed a full cosine cycle with $10\times$ learning rate decay. For each fixed FLOP count, we employ smoothed loss to determine the optimal model size with the smallest loss (Figure 2 (above)). Subsequently, we use Equation 1 and apply the `least_squares` method to fit the model.

Given the minimal variations in the final loss among a set of $(N, D)$ configurations, we classify these configurations as operating under "IsoLoss" conditions (see Appendix N Figure A16), considered optimal for training. In Figure 2 (below), we illustrate an efficient frontier interval that demonstrates permissible fluctuations in model size and dataset size at a specific FLOP count, while still achieving nearly identical losses. The variation in loss is quantified at 0.25 on a logarithmic scale with a base of 10. This indicates that within this FLOP counts, the model size can be adjusted within a range, increasing up to 80% or decreasing up to 40% without repeating data, to maintain a loss variation within 0.01.

We observe distinct growth rates in the proportional relationship between model size and training tokens for the MLM model compared to the CLM, as detailed in Table 2. Both models demonstrate an increase in the growth of model size that surpasses the growth of training tokens. Up to the intersection point around $1 \times 10^{22}$ (see Figure 2, left below), the model size of MLM tends to be smaller than the CLM, thereafter, the MLM rapidly exceeds that of the CLM. Notably, the growth of the MLM's training tokens is greatly lower than that for the CLM, possibly due to MLM's higher sample efficiency.

## 4. Experimental Validation

*Table 3.* **Model details.** We compare popular models PROGEN2 and ESM-2 using similar FLOPs with our models estimated by proposed scaling law.

| Params | Objective | Train. Tokens | FLOPs |
|---|---|---|---|
| PROGEN2-xlarge (6.4B) | CLM | 350B | $1.34 \times 10^{22}$ |
| Our 7.2B | CLM | 265B | $1.14 \times 10^{22}$ |
| ESM-2 (3B) | MLM | 1T | $1.68 \times 10^{22}$ |
| Our 10.7B | MLM | 260B | $1.68 \times 10^{22}$ |

Based on the scaling laws we observe, we estimate the model size and training tokens for current leading models by analyzing their FLOPs. The model's details are reported in Table 3.

### 4.1. Protein Generation Comparison: 7.2B CLM vs. 6.4B PROGEN-xlarge

We first evaluate the perplexity on OOD data and then compare the protein generation capabilities of the

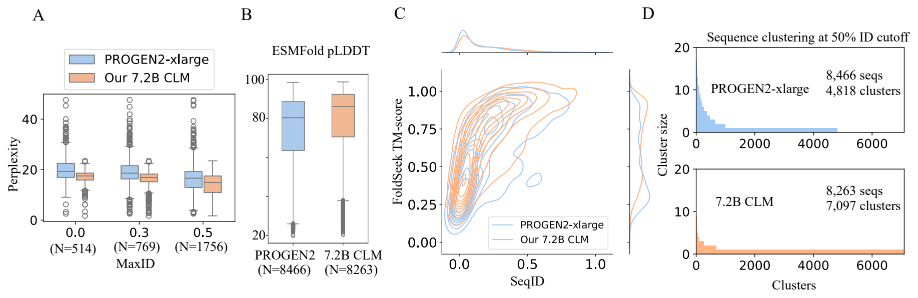

*Figure 3.* **Comparative Analysis of CLM Models. A.** Perplexity analysis for PROGEN2-xlarge and our 7.2B CLM shows lower values for our model across various MaxID levels. **B.** Box plots of pLDDT scores for protein structures by PROGEN2-xlarge and our 7.2B CLM. **C.** Contour and line plots show our 7.2B CLM sequences mimic natural sequences more closely than PROGEN2-xlarge. **D.** Clustering at 50% sequence identity reveals our 7.2B CLM generates more clusters.

7.2B CLM and PROGEN2-xlarge models. Each model generated 2,000 sequences for each parameter combination of top-$p$ $\{0.5, 0.7, 0.9, 1.0\}$ and temperature $t$ $\{0.2, 0.4, 0.6, 0.8, 1.0\}$, totaling 40,000 sequences per model. Sequences with a perplexity greater than 10 and duplicates were removed, leaving 8,263 and 8,466 sequences for the 7.2B CLM and PROGEN-xlarge, respectively. We used four metrics to assess the quality of the models and the generated sequences.

**OOD Dataset PPL Analysis** We randomly sampled 5,000 sequences from UniProt released after 2023-01-01 and aligned them to our and PROGEN2's training data (Uniref90 and BFD) using HHblits (Remmert et al., 2012) or Jackhmmer (European Bioinformatics Institute, n.d.). Sequences below a maximum identity cutoff were used to assess the models' PPL, as shown in Figure 3A. Our 7.2B CLM exhibited lower PPL on three subsets.

**pLDDT scores from ESMFold** Atomic structures of 8,263 and 8,466 generated sequences were predicted using ESMFold, and compared based on pLDDT scores, displayed in Figure 3B. The 7.2B model's average pLDDT score was 78.69, higher than PROGEN2-xlarge's 74.33.

**Natural Sequences Comparisons with Foldseek** Using Foldseek (van Kempen et al., 2022), we searched the PDB database for sequences similar to those generated by our 7.2B CLM model, which showed better mimicry of natural sequence properties with higher average TM-scores (0.655 vs 0.522) and SeqID (0.194 vs 0.165), as shown in Figure 3C.

**Diversity Analysis** Generated sequences were clustered using MMseqs2 (Steinegger & Söding, 2017) with a 50% similarity cutoff. The 7.2B CLM model resulted in higher diversity with 7,097 clusters compared to 4,818 clusters for PROGEN2-xlarge, detailed in Figure 3D.

*Table 4.* Tasks performance of MLM Model on the test dataset with LoRA fine-tuning.

| Models | Contact P. (P@L/5) | Fold C. (1195 class) | Fluor. R. (Reg.) |
|---|---|---|---|
| ESM-2 (3B) | 0.91 | 0.69 | 0.65 |
| Our 10.7B | 0.91 | **0.72** | **0.69** |

## 4.2. Protein understanding tasks: 10.7B MLM vs. 3B ESM2

We evaluate task types: Contact prediction as binary classification at the amino acid pair level; fold classification into 1195 classes at the sequence level; and fluorescence as regression tasks. We add a Multi-Layer Perceptron (MLP) head to each pre-trained model and apply Low-Rank Adaptation (LoRA) (Hu et al., 2021) for fine-tuning.

**Contact Prediction** This task determines if two residues, $i$ and $j$, are in contact based on a distance threshold (<8Å). Uses the trRosetta dataset (Du et al., 2021b), split into 12,041 training, 1,505 validation, and 1,505 test samples (Yang et al., 2020). The evaluation metric is P@L/5 accuracy, considering residue pairs with a separation length greater than 6 and a sequence length cutoff of 512. We find a similar performance with ESM-2 (3B).

**Fold Classification** This task assigns protein sequences to one of 1,195 known folds, primarily identifying novel remote homologs. This task is significant in proteomics and structural biology for analyzing folding patterns and advancing disease research (Chen et al., 2018). It shows improved accuracy over ESM-2 (3B), which we expect is due to increased data diversity scaling.

**Fluorescence** The fluorescence task predicts the fluorescence intensity of green fluorescent protein mutants. Following the TAPE splitting method (Rao et al., 2019), dataset sizes are 21.4K for training, 5.4K for validation, and 27.2K for testing. The evaluation metric is the Spearman score. Our 10.7B model, shows promising results on this task.

## References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bavarian, M., Jun, H., Tezak, N., Schulman, J., McLeavey, C., Tworek, J., and Chen, M. Efficient training of language models to fill in the middle. *arXiv preprint arXiv:2207.14255*, 2022.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

BFD Team. Big fantastic database. BFD Official Website, n.d. URL `https://bfd.mmseqs.com`.

Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Brandes, N., Ofer, D., Peleg, Y., Rappoport, N., and Linial, M. Proteinbert: a universal deep-learning model of protein sequence and function. *Bioinformatics*, 38(8):2102–2110, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020a.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020b.

Chen, B., Cheng, X., Li, P., Geng, Y.-a., Gong, J., Li, S., Bei, Z., Tan, X., Wang, B., Zeng, X., et al. xtrimopglm: unified 100b-scale pre-trained transformer for deciphering the language of protein. *arXiv preprint arXiv:2401.06199*, 2024.

Chen, J., Guo, M., Wang, X., and Liu, B. A comprehensive review and comparison of different computational methods for protein remote homology detection. *Briefings in bioinformatics*, 19(2):231–244, 2018.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Clark, A., de Las Casas, D., Guy, A., Mensch, A., Paganini, M., Hoffmann, J., Damoc, B., Hechtman, B., Cai, T., Borgeaud, S., et al. Unified scaling laws for routed language models. In *International conference on machine learning*, pp. 4057–4086. PMLR, 2022.

Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

Dauparas, J., Anishchenko, I., Bennett, N., Bai, H., Ragotte, R. J., Milles, L. F., Wicky, B. I., Courbet, A., de Haas, R. J., Bethel, N., et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378 (6615):49–56, 2022.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A. P., Caron, M., Geirhos, R., Alabdulmohsin, I., et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pp. 7480–7512. PMLR, 2023.

Delétang, G., Ruoss, A., Duquenne, P.-A., Catt, E., Genewein, T., Mattern, C., Grau-Moya, J., Wenliang, L. K., Aitchison, M., Orseau, L., et al. Language modeling is compression. *arXiv preprint arXiv:2309.10668*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., and Tang, J. Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2103.10360*, 2021a.

Du, Z., Su, H., Wang, W., Ye, L., Wei, H., Peng, Z., Anishchenko, I., Baker, D., and Yang, J. The trrosetta server for fast and accurate protein structure prediction. *Nature protocols*, 16(12):5634–5651, 2021b.

Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., and Tang, J. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 320–335, 2022.

Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., 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, 2021.

Elnaggar, A., Essam, H., Salah-Eldin, W., Moustafa, W., Elkerdawy, M., Rochereau, C., and Rost, B. Ankh: Optimized protein language model unlocks general-purpose modelling. *arXiv preprint arXiv:2301.06568*, 2023.

European Bioinformatics Institute. Jackhmmer tool. EBI Tools Documentation, n.d. URL https://www.ebi.ac.uk/Tools/hmmer/search/jackhmmer.

fast.ai. How could the memorization hypothesis be true. fast.ai Blog, 2023. Retrieved May 21, 2024, from https://www.fast.ai/posts/2023-09-04-learning-jumps/how-could-the-memorization-hypothesis-be-true.

Ferruz, N., Schmidt, S., and Höcker, B. Protgpt2 is a deep unsupervised language model for protein design. *Nature communications*, 13(1):4348, 2022.

Heinzinger, M., Weissenow, K., Sanchez, J. G., Henkel, A., Steinegger, M., and Rost, B. Prostt5: Bilingual language model for protein sequence and structure. *bioRxiv*, pp. 2023–07, 2023.

Henighan, T., Kaplan, J., Katz, M., Chen, M., Hesse, C., Jackson, J., Jun, H., Brown, T. B., Dhariwal, P., Gray, S., et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

Hernandez, D., Kaplan, J., Henighan, T., and McCandlish, S. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

Hernandez, D., Brown, T., Conerly, T., DasSarma, N., Drain, D., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Henighan, T., Hume, T., et al. Scaling laws and interpretability of learning from repeated data. *arXiv preprint arXiv:2205.10487*, 2022.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jacobs, S. A., Tanaka, M., Zhang, C., Zhang, M., Song, L., Rajbhandari, S., and He, Y. Deepspeed ulysses: System optimizations for enabling training of extreme long sequence transformer models. *arXiv preprint arXiv:2309.14509*, 2023.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Komatsuzaki, A. One epoch is all you need. *arXiv preprint arXiv:1906.06669*, 2019.

Levy Karin, E., Mirdita, M., and Söding, J. Metaeuk—sensitive, high-throughput gene discovery, and annotation for large-scale eukaryotic metagenomics. *Microbiome*, 8:1–15, 2020.

Li, F.-Z., Amini, A. P., Yue, Y., Yang, K. K., and Lu, A. X. Feature reuse and scaling: Understanding transfer learning with protein language models. *bioRxiv*, pp. 2024–02, 2024.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., dos Santos Costa, A., Fazel-Zarandi, M., Sercu, T., Candido, S., and Rives, A. Evolutionary-scale prediction of atomic level protein structure with a language model. *bioRxiv*, 2022.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023.

Liu, H., Zaharia, M., and Abbeel, P. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023a.

Liu, X., Yan, H., Zhang, S., An, C., Qiu, X., and Lin, D. Scaling laws of rope-based extrapolation. *arXiv preprint arXiv:2310.05209*, 2023b.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

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.

Markowitz, V. M., Korzeniewski, F., Palaniappan, K., Szeto, E., Werner, G., Padki, A., Zhao, X., Dubchak, I., Hugenholtz, P., Anderson, I., et al. The integrated microbial genomes (img) system. *Nucleic acids research*, 34 (suppl_1):D344–D348, 2006.

McCandlish, S., Kaplan, J., Amodei, D., and Team, O. D. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in neural information processing systems*, 34:29287–29303, 2021.

Merrill, W., Ramanujan, V., Goldberg, Y., Schwartz, R., and Smith, N. Effects of parameter norm growth during transformer training: Inductive bias from gradient descent. *arXiv preprint arXiv:2010.09697*, 2020.

Muennighoff, N., Rush, A., Barak, B., Le Scao, T., Tazi, N., Piktus, A., Pyysalo, S., Wolf, T., and Raffel, C. A. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Nguyen, E., Poli, M., Durrant, M. G., Thomas, A. W., Kang, B., Sullivan, J., Ng, M. Y., Lewis, A., Patel, A., Lou, A., et al. Sequence modeling and design from molecular to genome scale with evo. *bioRxiv*, pp. 2024–02, 2024.

Nijkamp, E., Ruffolo, J. A., Weinstein, E. N., Naik, N., and Madani, A. Progen2: exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978, 2023.

Notin, P., Kollasch, A., Ritter, D., Van Niekerk, L., Paul, S., Spinner, H., Rollins, N., Shaw, A., Orenbuch, R., Weitzman, R., et al. Proteingym: large-scale benchmarks for protein fitness prediction and design. *Advances in Neural Information Processing Systems*, 36, 2024.

PyTorch Lightning. Learning rate finder. PyTorch Lightning Documentation, n.d. URL https://pytorch-lightning.readthedocs.io/en/1.5.10/advanced/lr_finder.html.

Qiu, J., Xu, J., Hu, J., Cao, H., Hou, L., Gao, Z., Zhou, X., Li, A., Li, X., Cui, B., et al. Instructplm: Aligning protein language models to follow protein structure instructions. *bioRxiv*, pp. 2024–04, 2024.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21 (140):1–67, 2020.

Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, X., Canny, J., Abbeel, P., and Song, Y. S. Evaluating protein transfer learning with tape. In *Advances in Neural Information Processing Systems*, 2019.

Remmert, M., Biegert, A., Hauser, A., and Söding, J. Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment. *Nature methods*, 9(2):173–175, 2012.

Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., Keysers, D., and Houlsby, N. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34: 8583–8595, 2021.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Shazeer, N. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Steinegger, M. and Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature biotechnology*, 35(11):1026–1028, 2017.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and Consortium, U. Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932, 2015.

Tay, Y., Dehghani, M., Rao, J., Fedus, W., Abnar, S., Chung, H. W., Narang, S., Yogatama, D., Vaswani, A., and Metzler, D. Scale efficiently: Insights from pretraining and fine-tuning transformers. *arXiv preprint arXiv:2109.10686*, 2021.

Tay, Y., Dehghani, M., Tran, V. Q., Garcia, X., Wei, J., Wang, X., Chung, H. W., Shakeri, S., Bahri, D., Schuster, T., et al. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*, 2022.

Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., and Stojnic, R. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*, 2022.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

van Kempen, M., Kim, S. S., Tumescheit, C., Mirdita, M., Gilchrist, C. L., Söding, J., and Steinegger, M. Foldseek: fast and accurate protein structure search. *Biorxiv*, pp. 2022–02, 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Verkuil, R., Kabeli, O., Du, Y., Wicky, B. I., Milles, L. F., Dauparas, J., Baker, D., Ovchinnikov, S., Sercu, T., and Rives, A. Language models generalize beyond natural proteins. *bioRxiv*, pp. 2022–12, 2022.

Wang, A. and Cho, K. Bert has a mouth, and it must speak: Bert as a markov random field language model. *arXiv preprint arXiv:1902.04094*, 2019.

Wang, H., Ma, S., Dong, L., Huang, S., Zhang, D., and Wei, F. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Wang, T., Roberts, A., Hesslow, D., Le Scao, T., Chung, H. W., Beltagy, I., Launay, J., and Raffel, C. What language model architecture and pretraining objective works best for zero-shot generalization? In *International Conference on Machine Learning*, pp. 22964–22984. PMLR, 2022.

Yang, J., Anishchenko, I., Park, H., Peng, Z., Ovchinnikov, S., and Baker, D. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., et al. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*, 2022.

Zhai, X., Kolesnikov, A., Houlsby, N., and Beyer, L. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12104–12113, 2022.

Zhang, B., Liu, Z., Cherry, C., and Firat, O. When scaling meets llm finetuning: The effect of data, model and finetuning method. *arXiv preprint arXiv:2402.17193*, 2024.

Zheng, Z., Deng, Y., Xue, D., Zhou, Y., Ye, F., and Gu, Q. Structure-informed language models are protein designers. In *International Conference on Machine Learning*, pp. 42317–42338. PMLR, 2023.

## A. Conclusion

We first expanded metagenomic databases, emphasizing the critical importance of data quality and quantity for scaling protein language models. Then we introduced scaling laws for two distinct language model tasks and provided recommendations on the allocation of model size and data size under expanding compute budget, along with the potential loss prediction. Additionally, we explored the transferability between these tasks. Our findings were validated by robust performance across various downstream tasks. We expect that they will extend to and assist other domains and biological data modalities.

## B. Related Work

**Protein Language Model** Since the advent of AlphaFold2 (Jumper et al., 2021), the masked language model (MLM) has been integrated as a subtask within the Evoformer architecture. In this context, an assumption is that large language models can be considered as a lossless compression method (Delétang et al., 2023). This was followed by a series of language modeling efforts (Ferruz et al., 2022; Brandes et al., 2022; Heinzinger et al., 2023; Elnaggar et al., 2021; 2023), which aimed to conduct pre-training on single-sequence proteins using larger datasets and model scales. These efforts sought to harness the scale of the models to learn complex co-evolutionary information, although detailed investigations on how to optimally scale these models remain scarce. Our work primarily focuses on these finer aspects, aiming to fill this gap in the research.

**Training objectives** In natural language processing (NLP), masked language models (MLM) are rarely adopted due to the self-explanatory nature of natural language, which inherently prompts the meta-knowledge of tasks and generates task targets through CLM (Conditional Language Modeling) training models. However, a unified language modeling objective for Protein Language Models has yet to be fully consented. Those based on causal language modeling (CLM) have been primarily explored for protein design. Benchmarks in protein design using MLM (Wang & Cho, 2019) have also shown promising results for generation (Notin et al., 2024), exhibiting variable performance when compared to CLM (Zheng et al., 2023; Verkuil et al., 2022). Additionally, the potential of the in-filling task objective remains largely unexplored (Bavarian et al., 2022; Tay et al., 2022; Du et al., 2021a). Our research aims to thoroughly discern the scaling behavior of the two most common optimization objectives in this domain.

**Scaling Laws** To our knowledge, the concept of scaling laws of language model is first introduced by OpenAI (Kaplan et al., 2020). Subsequently, numerous variants and modifications (Hoffmann et al., 2022) have been developed around this theme. Recently, an array of new scaling laws has emerged. These include scaling laws related to learning rates and batch sizes (Bi et al., 2024), data-constrained scaling laws (Muennighoff et al., 2024), scaling laws for downstream tasks and Transfer (Zhang et al., 2024; Hernandez et al., 2021), as well as scaling laws within the Mixture of Experts (MoE) framework (Clark et al., 2022), and those concerning long sequences and positional encoding (Liu et al., 2023b). While these laws are primarily derived using auto-regressive models in resource-rich domains, their application in the biological data sector is less common. Our work seeks to address this gap. Furthermore, scaling laws for Masked Language Models (MLM) are notably scarce. Given that MLMs are currently one of the most effective training methods for biological data, our research on MLMs could also be extended to other non-text domains.

## C. Discussion and Limitations

**Repetition for Masked Language Model** Our scaling law is learned within a single epoch setting. It is well known that MLM exhibits higher sample efficiency than CLM due to the variable masking of training samples across epochs. However, this advantage diminishes when training is limited to only one epoch. This also suggests that for MLM training, a small amount of repetition can be considered as new data, without detriment to the performance. We present empirical evidence comparing a 2.8B model trained on 1T tokens (approximately five epochs) against a 10.7B model trained on 265B tokens (roughly 1.4 epochs). Despite the models utilizing the same amount of FLOPs, the latter achieves optimal training by attaining lower out-of-distribution (OOD) perplexity (10.33 vs 10.21). Despite this, the impact of training MLM for several epochs repeatedly is not significant in terms of loss. This insight suggests that repeating several rounds under MLM training has a minimal impact on reducing loss, and our scaling law does not necessarily need to be confined within 200 billion tokens. And smaller models are more user-friendly during inference and fine-tuning. Therefore, we also suggest an alternative approach that adjusts the optimal training token count and model size within our scaling law framework when scaling MLM. We will further investigate repeat scaling laws as designated in future work (Muennighoff et al., 2024).

**Multi-modality Scaling** We observe that the scaling laws for CLM, also known as autoregressive models, exhibit similarities to those in natural languages or the code modality in the context of protein sequences, closely aligning with findings by Chinchilla (Hoffmann et al., 2022). The multi-modal auto-regressive work (Henighan et al., 2020) suggests the existence of a nearly universal scaling law across various modalities, including images, videos, math, code, and languages. Our results appear in this trend as well. The same situation may apply to other modalities of biological data, such as RNA and DNA (Nguyen et al., 2024). However, for the scaling laws pertaining to MLM, we have yet to identify detailed research on or evidence of a universal scaling law. This gap highlights a crucial area for future investigations, potentially extending our understanding of scaling across different model architectures and data modalities.

**Hyperparameters Sensitivity** When selecting language models and configuring their training processes, model size and number of training tokens are not the only hyperparameters that require careful selection. Other critical factors, such as the learning rate schedule, batch size, also play significant roles. We rely on existing work and provided experimental heuristics to determine the other necessary hyperparameters (Zeng et al., 2022; Touvron et al., 2023b). The maximal learning rate (LR) was determined using a LR finder (PyTorch Lightning, n.d.) to prevent model collapse, then empirically choosing the median value where the loss decline is steepest. Our observations suggest that while the exact maximum learning rate was not highly sensitive, ensuring a steep decline in loss and the completion of the learning rate schedule was crucial, typically with an error loss of around 0.01. The critical batch size (McCandlish et al., 2018) and model loss are correlated (Kaplan et al., 2020), indicating that larger models require correspondingly larger batch sizes; we empirically set models under 1B parameters to a batch size of 512K and models over 1B to 1 million.

**Other Dataset and Strategies** While our datasets encompass a significant portion of the protein universe, they might still not be fully representative. The combination of BFD (BFD Team, n.d.), Uniref (Suzek et al., 2015), MetaClust (Levy Karin et al., 2020), and IMG/JGI (Markowitz et al., 2006) with 90% clustering includes at least 600B unique tokens. However, different datasets might induce variations in the power-law behavior, such as changes in the slope or shifts in the log-log space, which warrants further investigation. It may be of interest for future work to test the applicability of our findings to different model architectures. Currently, there is significant research into the scaling of LLMs for long sequences (Choromanski et al., 2020; Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020; Dao et al., 2022; Liu et al., 2023a; Jacobs et al., 2023), and the MSA augmentation could notably enhance protein representation in terms of contacts and structure. Exploring the scaling laws in this context may be an interesting avenue for future work.

## D. UR50/S Repeat Experiments

We employed three different methods to repeat training on the UR50/S dataset, all of which ultimately led to overfitting. The reference for these experiments is shown by the blue curve in Figure A4, which represents UniMeta's loss for approximately one epoch.

Firstly, using bootstrapping, we processed 200 billion tokens from UR50/S with replacement. In each epoch, 65% of the dataset was randomly selected, leading to a diminished proportion of unsampled tokens by the fifth epoch, as depicted by the orange curve.

Secondly, we shuffled the unique data for each epoch to ensure that all UR50/S tokens were used per epoch, resulting in a stair-step pattern (fast.ai, 2023) in the training loss, illustrated by the green curve. It has simply memorized the dataset but isn't improving at generalizing. Over-confident predictions of the first batch of the next epoch lead to a big step update, and then the model is not adapted to the next batches, resulting in no longer a decrease in loss.

Lastly, we shuffled the entire training dataset less stringently, which did not strictly ensure that all UR50/S tokens were used every epoch, but guaranteed that each token was used an equal number of times over the entire training period. We term it global shuffle, this approach is shown by the red curve..

From the gradient norm curve shown in Figure A4 (right), we observe an uptick in gradient norm for the overfitting curves, indicating that the model is no longer optimizing effectively. In machine learning, such an increase in gradient norm typically suggests that the model is encountering areas of the parameter space where gradients are steeper or more erratic, often occurring when the model starts to memorize the training data rather than generalize from it, approaching a saturated network (Merrill et al., 2020). This behavior can result from overly complex models, too many training epochs without sufficient regularization, or training on non-representative data.

## E. Choice of Masking Ratio

In the original BERT work (Devlin et al., 2018), the absence of masked tokens in downstream tasks presented a mismatch with the pre-training data distribution. The authors investigated various masking ratios and concluded that a 15% masking rate was most beneficial for downstream tasks. This was implemented alongside an 80-10-10 strategy: 80% of the tokens were replaced with a mask, 10% were randomly substituted, and the remaining 10% were left unchanged.

However, given the significant differences between protein sequences and natural language processing data, we em-

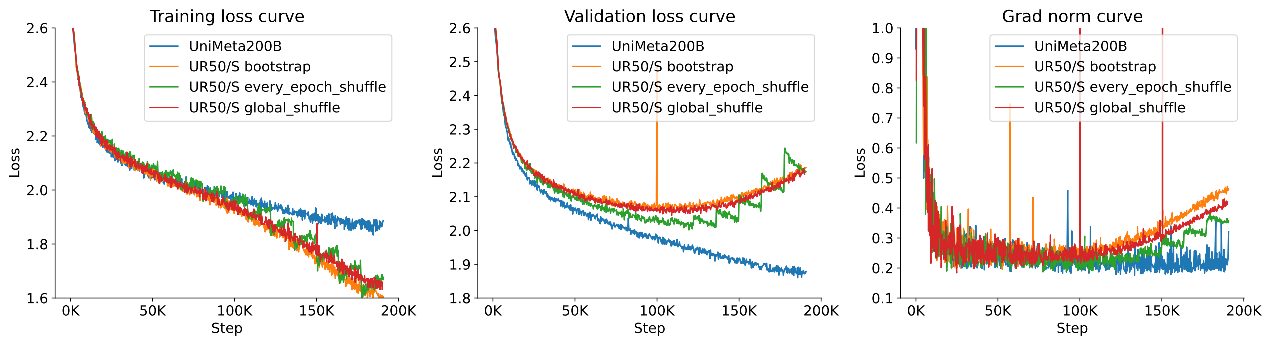

*Figure A4.* **Learning curve for UR50/S dataset repetition methods**. Our 194B tokens dataset (UniMeta200B) shown in blue, serves as the reference with an approximate single epoch run. The bootstrapping method, depicted in orange, processes 200 billion tokens with replacement, indicating a tendency towards zero unsampled tokens by the fifth epoch. The every-epoch shuffle method, in green, ensures all tokens are used per epoch, forming a stair-step pattern in training loss. Lastly, the global shuffle method, in red, loosely uses all tokens each epoch but ensures the strict number of epoch passes for every token. The rightmost plot of gradient norms shows an uptick for curves corresponding to overfitting, signifying a lack of further optimization, with steep or erratic gradients indicated by the ascending gradient norms.

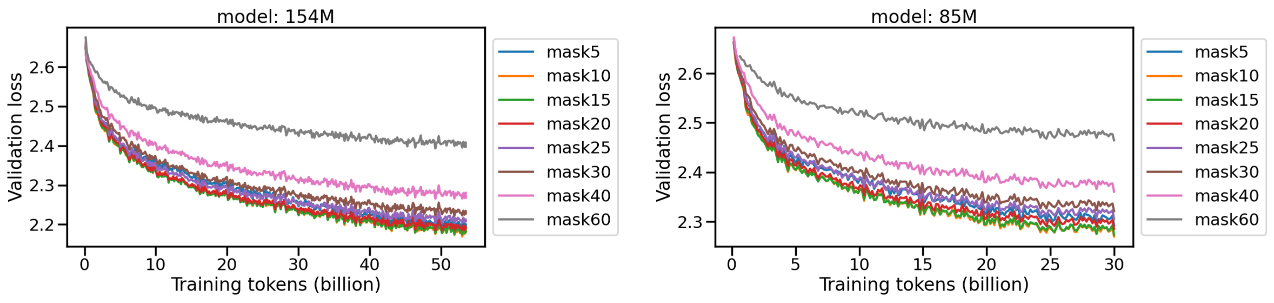

*Figure A7.* **Validation loss of different masking ratios**. Two models (154M and 85M) are trained from 5% to 60% masking intervals.

ployed two models, sized at 85M and 154M, to explore a range of masking ratios from 5% to 60% (see Figure A7). The best masking ratios for validation loss drop ranged from 10% to 20%; ratios too small (5%) or too large (greater than 25%) degraded the performance.

We further used pre-trained eight different models to perform full fine-tuning on downstream tasks such as Contact Prediction and Fold Classification in Figure A8. Results from the test datasets revealed that, similar to NLP, the optimal performance was achieved within a 10%-20% masking range. Specifically, a 20% masking ratio slightly outperformed 15% in Contact Prediction, while the 15% ratio yielded the best results in Fold Prediction. Consequently, for our Masked Language Model (MLM), we decided to adhere to the 15% masking ratio with the 80-10-10 strategy for training all our models.

## F. MLM/CLM for Protein Contact Prediction

We compared the effectiveness of CLM in the downstream task of contact prediction, using two different setups (Figure A9). In the first setup, two 3B models were trained under identical computational resources on 200 billion tokens, $3.4 \times 10^{21} FLOPs$. Their performance was evaluated through two training approaches: Probing (freezing the pre-trained model) and LoRA fine-tuning, with an added MLP head for comparison.

In the second setup, we compared the effects of MLM and CLM under similar loss conditions. Here, a 7.2B CLM model and an 880M MLM model were selected, both achieving a loss of 1.98 on our validation set. Despite the MLM model having a simpler loss calculation, involving a 15% mask rather than a one-by-one mask—which would result in a higher loss—the MLM significantly outperformed the CLM. Importantly, the CLM model's computational power was an order of magnitude greater than the MLM model ($1.68 \times 10^{22}$ vs $1.0 \times 10^{21}$ FLOPs). This suggests that despite the lower loss achievable by the CLM model com-

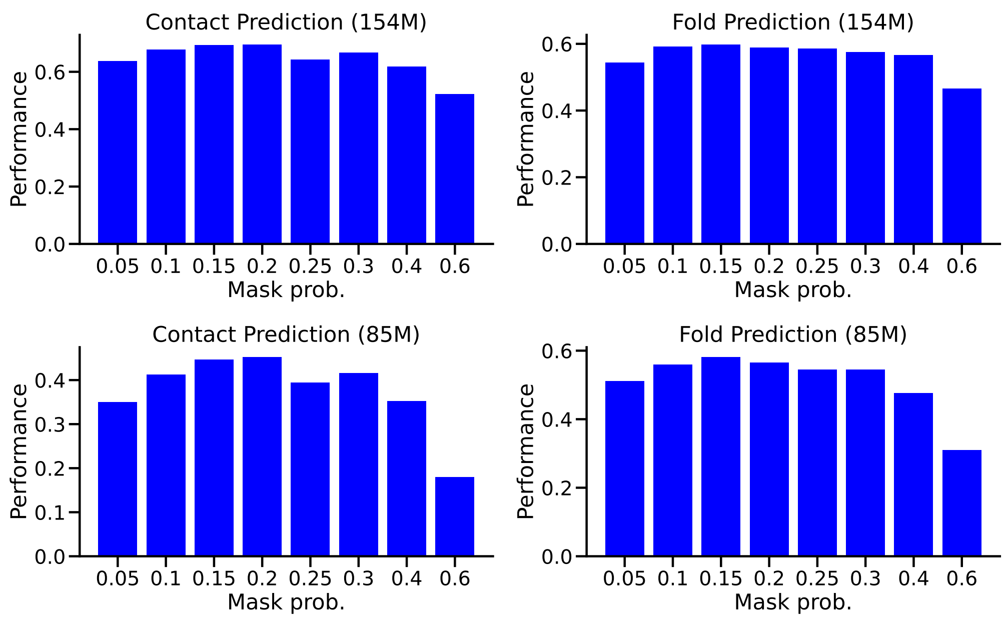

*Figure A8.* **Abalation of different masking ratios**. Two models (154M and 85M) are trained from 5% to 60% masking intervals, and evaluated on contact map and fold classification downstream tasks.

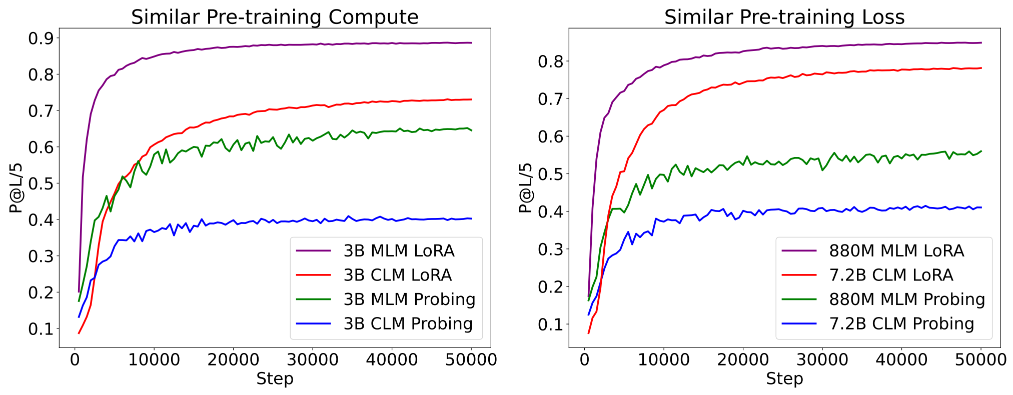

*Figure A9.* **Contact Prediction on MLM and CLM models.** Two 3B models (CLM and MLM) were trained using identical computational resources, represented by the probing and LoRA fine-tuning methods. On the right, performance of a 7.2B CLM model is compared with an 880M MLM model under similar pre-training loss conditions. These models exhibit differing rates of convergence, highlighting the impact of uni-directional and bi-directional model architectures on learning dynamics.

pared to MLM with a one-by-one mask, the unidirectional limitations of CLM do not translate into better downstream task performance.

## G. Pre-training Dataset Quality

Compared to Uniref90, ColabFoldDB offers a higher diversity and larger numbers of protein sequences, though with generally shorter sequence lengths, likely suggesting potentially lower data quality. To evaluate the efficacy of our expanded dataset, ColabFoldDB, we initially trained two 85M models separately on Uniref90 and ColabFoldDB. Uniref90 in our dataset comprises two subsets: Uniref50/S and the incremental dataset over Uniref50/S, termed Uniref90/50. Similarly, ColabFoldDB consists of representative and member data. We controlled the sampling proportion to ensure uniform sampling across both datasets, with results reported in Table A5. Both models were then trained using identical configurations on a 50B scale.

From the perspective of validation loss in pre-training, the higher loss on ColabFoldDB might be attributed to its lower diversity and shorter sequence lengths compared to Uniref90. However, the performance on downstream tasks, such as contact prediction and fold classification, shows negligible differences between models trained solely on ColabFoldDB and those trained on Uniref90, as illustrated in Figure G. This confirms that ColabFoldDB is an effective expansion of Uniref90 that maintains sample efficiency.

*Table A5.* **Compared two dataset characteristics.** Protein sequence count, token number, and sampling proportions for Uniref50/S, Uniref90/50, and ColabFoldDB representative and member data.

| Datasets | Prot. Seq. | Tokens (AAs) | Sampling Prop. |
|---|---|---|---|
| Uniref50/S | 54M | 15.2B | 28.67% |
| Uniref90/50 | 102M | 37.8B | 71.33% |
| ColabFoldDB$_c$ | 208M | 37.7B | 26.75% |
| ColabFoldDB$_m$ | 575M | 103B | 73.52% |

## H. Convergence of Downstream Fine-tuning Tasks

Observing the learning curves in Figure A11, we can assess the effectiveness of different fine-tuning scenarios. For the contact prediction task, the convergence speed under the LoRA setting is very similar for both models. Our testing reveals closely matching results for ESM-2 models with capacities of 650M, 3B, 15B, consistent with the findings reported by Ankh et al. (Elnaggar et al., 2023). This similarity suggests possible saturation of the dataset under single-sequence pre-trained models. Additionally, the convergence rates for tasks such as fold classification and fluorescence are generally faster than those for ESM-2, indicating robust generalization following our data augmentation strategies.

## I. Mixed Objectives Training

We also employed an untied model to simultaneously optimize two objectives:

$$L_{CLM} = \text{CE}(V\sigma(W_1(\text{ encoder}(x))), y_{\text{next}}),$$

and

$$L_{MLM} = \text{CE}(V\sigma(W_2(\text{encoder}(x))), y_{\text{mask}}),$$

where $V$ represents the protein vocabulary embedding, and $W_1$ and $W_2$ are the parameters corresponding to the CLM and MLM tasks, respectively. `CE` is the cross-entropy operator. The $\sigma$ is the `Tanh` activation function.

We compared CLM and MLM under our scaling law of optimal model and data size distributions. One approach involved training from scratch, while the other used mixed training. In the mixed training approach, the actual number of training tokens was higher due to the additional FLOPs consumed by another optimally trained objective, in other words. In other words, mixed training consumes the FLOPs of two optimal allocations; we only extracted the loss curve of one target for comparison. We extracted the loss curve of just one target for comparison with the from-scratch training. Our findings indicate that mixed training of the two targets can lead to detrimental interference, an effect not observable in smaller models, as depicted in Figure A12. As the model size increases to hundred million or billion parameters, the differences become more pronounced. Therefore, if both objectives are to be optimized concurrently, a sequential training strategy should be employed: first optimizing CLM, followed by MLM training. We consider that CLM is more challenging to predict than MLM, which may allow the model to capture more complex and implicit sequential features initially, thereby enhancing its ability to understand and predict masked words in subsequent MLM training.

## J. Loss Prediction

In exploring the scaling relations of loss, we analyzed various model sizes $N$, compute budgets $C$, and training dataset tokens $D$. These can be described by a similar power-law relation defined as:

$$L(x) = \beta_x \times x^{\alpha_x} \tag{2}$$

where $\alpha_x$ is the scaling exponent for different variables. For each FLOP count, we aimed to identify the minimal loss as the fitting target along with the corresponding independent variable $x$. Table A6 presents these fitting coefficients.

*Table A6.* Coefficient of Equation 2

| Objective | $\alpha_N$ | $\alpha_D$ | $\alpha_C$ | $\beta_N$ | $\beta_D$ | $\beta_C$ |
|---|---|---|---|---|---|---|
| CLM | $-0.037$ | $-0.051$ | $-0.027$ | $4.835$ | $7.904$ | $8.251$ |
| MLM | $-0.040$ | $-0.120$ | $-0.034$ | $4.530$ | $42.614$ | $10.125$ |

Based on the coefficients obtained from the fitting described above, we can establish the relationship between $D$ and $N$ by eliminating $L$. The relationship is expressed by the following equation:

$$D(N) = \left(\frac{\beta_N}{\beta_D}\right)^{\frac{1}{\alpha_D}} \times N^{\frac{\alpha_N}{\alpha_D}} \tag{3}$$

By substituting the learned coefficients into this formula, we can derive $D^{\text{opt}}_{\text{MLM}}$ and $D^{\text{opt}}_{\text{CLM}}$ when given $N$. The estimation may be affected when the data exceeds 200 billion or when the quality or quantity of the training dataset changes.

Following both individual power-laws, it is possible to integrate two independent scaling laws (see Appendix O) and allocate two FLOPs within a specified compute budget to train two optimal models if our goal is to simultaneously

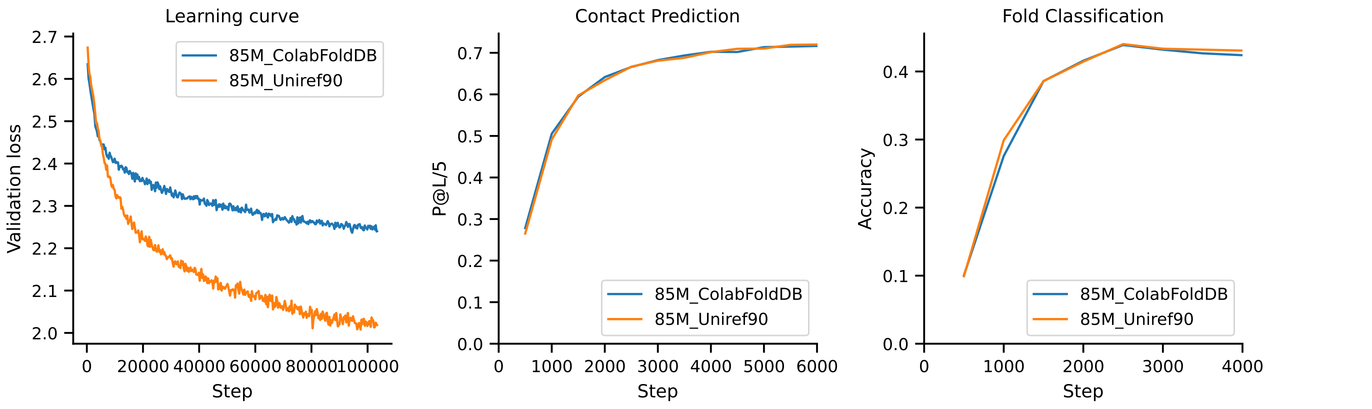

*Figure A10.* **Data quality check.** Comparison of learning dynamics and downstream task performance for two 85M models trained on ColabFoldDB and Uniref90. Left: Validation loss curves demonstrating initial training differences. Middle: Contact prediction performance showing the response to testing on similar tasks. Right: Fold classification accuracy, comparing model responses to structural prediction tasks. Despite initial differences in loss, both datasets yield comparable performance in downstream applications.

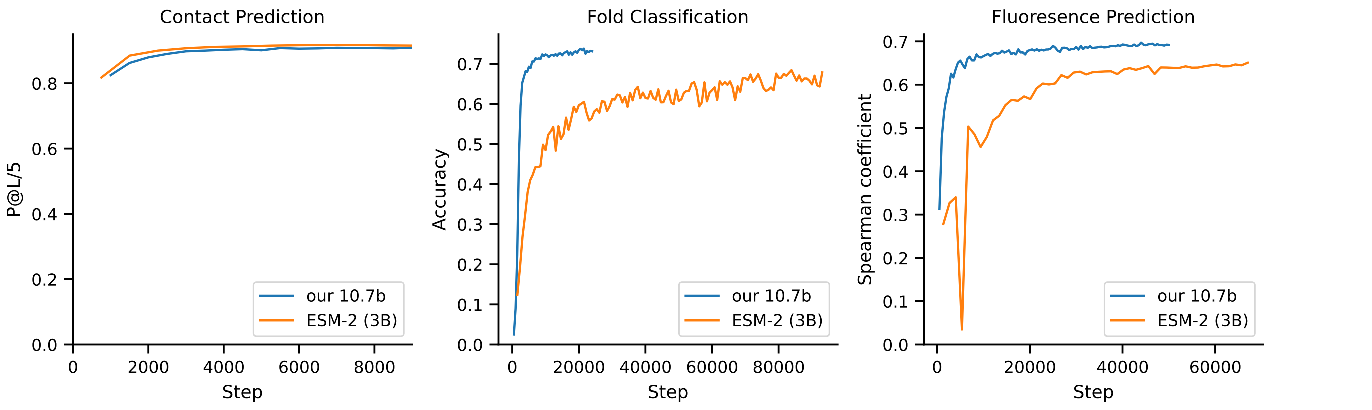

*Figure A11.* **Learning Curve Convergence.** LORA fine-tuning our 10.7B model and ESM-2 (3B) model on three downstream tasks.

obtain both optimal training models. We further find that the scaling behavior of sparse parameter counts in a Mixture of Experts (MoE) model, set with eight experts (see Appendix L), as well as a combined power-law formula used to fit our data (see Appendix M), both exhibit a certain similarity to the scaling behavior we have proposed.

## K. Transfer Scaling

We have outlined two independent scaling laws and how to allocate FLOPs under a fixed budget for training two optimal models, one with MLM and the other with CLM. However, we have not explored the interaction between these objectives. This raises important questions: Can models trained with one objective transferred to one with another objective? Is there a synergistic effect from training two models? Does training order impact the results?

### K.1. Transferability

We conduct transfer learning experiments on MLM and CLM objectives, selecting eight optimal model sizes based on Equation 1. These models correspond to four increasing FLOP counts from $3\times10^{19}$ to $1\times10^{21}$ and undergo training from scratch followed by transfer training. Transfer training involves initially training on MLM or CLM, then training on the alternate model for each size.

We find that optimal pre-training on one objective benefits the target objective in transfer learning, though effects vary between methods. Starting with CLM and then training MLM, benefits increase with model scale. In contrast, starting with MLM then training CLM sees diminishing benefits. As shown in Figure A13 (left), for a model size of 230M with $3\times10^{19}$ FLOPs, MLM from CLM pre-training reduces the loss by 0.02 compared to MLM from scratch, however,

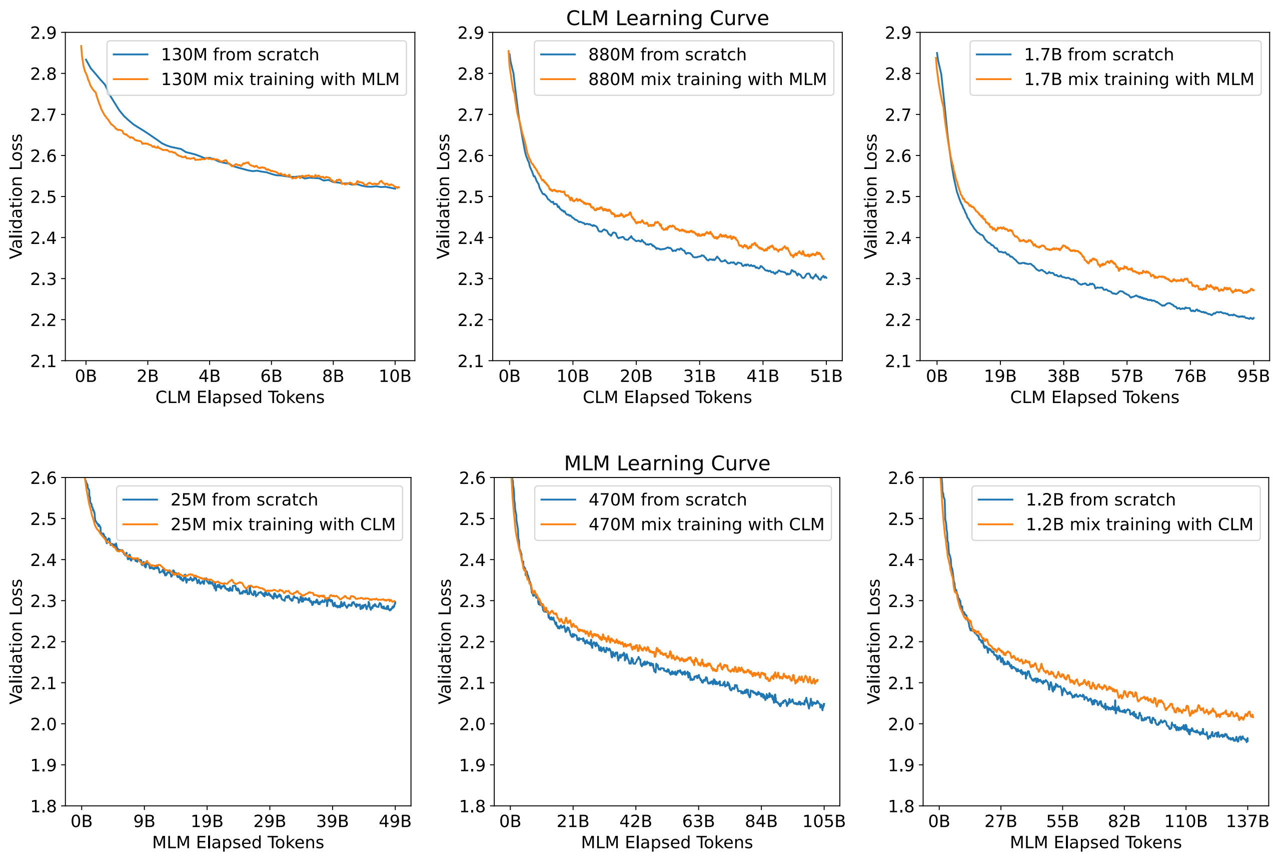

*Figure A12.* **Mixed objective validation loss**. Comparative validation loss curves for models trained from scratch versus mixed training approaches. Each panel corresponds to different model sizes, as indicated by the parameters. For each model, two training strategies were compared over an identical number of elapsed tokens: training from scratch (blue) and mixed training with the other objective (orange). Across all model sizes, training from scratch consistently achieves lower validation loss compared to mixed training, suggesting that mixed training may not be as effective as dedicated training for each individual objective.

benefit that nears zero for the 1.7B model. Conversely, for models from 85M to 1.2B, transfer benefits grow with model size, the compared validation loss gap increasing from 0.025 to 0.045. This likely stems from the higher loss utilization rate in MLM; CLM calculates losses for all tokens in a protein sequence, whereas MLM only calculates losses for 15% of the tokens. [2].

We use a power-law to model the transfer scaling law, initially excluding the pre-training FLOPs. The scaling behavior of transfer learning is modeled by:

$$L(C_s) = A_s \times C_s^{\alpha_s}, \quad L(C_t) = B_t \times C_t^{\alpha_t} \qquad (4)$$

where $L(C_t)$ and $L(C_s)$ represent the loss for transfer learning and training from scratch.

Figure A14 (right) shows that the efficient frontier for $L(C_t)$ has shifted relative to $L(C_s)$ (it can be directly obtained from Table A6, repeated here for convenience.), indicating an improvement. The coefficients from both are shown in Table A7, where we can infer that $C_t \propto C_s^{\frac{\alpha_s}{\alpha_t}} = C_s^{0.89}$, suggesting that training MLM from scratch with 10× the compute requires approximately 7.7× the compute compared to MLM from CLM pre-training. Another observation is that mixing training objectives in a single batch tends to be detrimental. Detailed results and settings are in Appendix I. The recommended transfer learning schedule involves pre-training CLM before MLM, as mixed training and order swapping show no benefits. We speculate that this primarily occurs because our MLM, which focuses solely on recovering corruption tokens, is not causal. If it predicted a middle segment in a left-to-right manner, it could mutually adapt

*Table A7.* Coefficients for $L(C_s)$ and $L(C_t)$

| Parameter | $A_s$ | $\alpha_s$ | $B_t$ | $\alpha_t$ |
|---|---|---|---|---|
| MLM | 10.124 | −0.034 | 11.133 | −0.038 |
| CLM | 8.142 | −0.027 | 7.191 | −0.024 |

[2]Appendix E analyzes the mask ratios.

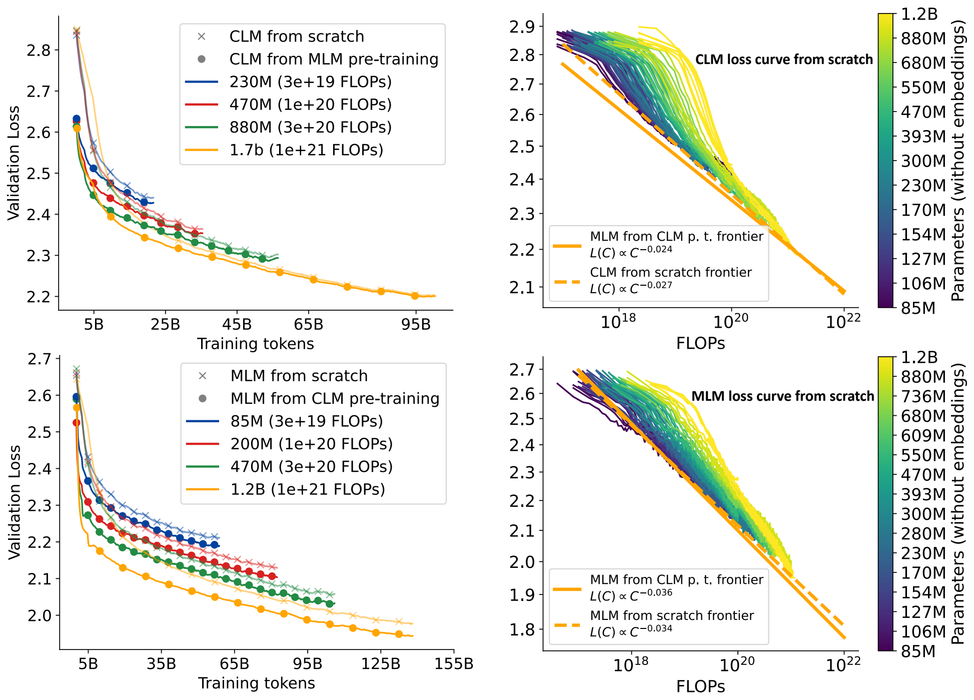

*Figure A13.* **Left:** The upper graph compares validation loss of CLM trained from scratch with those transferred from MLM, showing diminishing transfer benefits as model size increases. The lower graph depicts increased benefits for MLM from pre-trained CLM with larger sizes, indicating scale-dependent efficiency gains. **Right:** Shows loss curves for CLM and MLM across different FLOPs, emphasizing the efficient frontiers (or Pareto Frontier) from various transfer strategies. It highlights that the benefits of transferring from CLM to MLM grow with model size, reflecting a scale-dependent synergy between training objectives.

with the context to accelerate training (Wang et al., 2022).

## K.2. Effectively Transferred Tokens

Although we observe that MLM benefits from transfer learning from CLM, the pre-training compute budget remains unaccounted for. We focus on two aspects: (1) the actual benefit CLM provides to MLM and its predictability, and (2) performance differences between MLM trained from pre-trained CLM (MLM-CLM) and MLM from scratch (MLM-S) under identical FLOP constraints. We define Effectively Transferred Tokens $D_t$ as the *additional data a model of the same size would need to train from scratch on MLM to achieve the same loss as a model pre-trained on CLM*. If the token number in the pre-trained CLM model exceeds $D_t$, then the computations for CLM pre-training were excessive. Knowing $D_t$ in advance would guide the allocation of tokens for CLM pre-training.

We compare MLM-S and MLM-CLM models ranging from 33M to 1.2B with FLOP counts from $3 \times 10^{19}$ to $1 \times 10^{21}$. By calculating the *token distance* at the same loss level between these models, we establish our fitting target $D_t$, collecting approximately 2800 sample points. Following similar methods in scaling transfer works (Henighan et al., 2020; Zhang et al., 2024), $D_t$ is defined by a simple multiplicative scaling formula:

$$D_t = k \times \frac{1}{D_f^{\delta}} \times \frac{1}{N^{\gamma}},$$

where coefficients $k \approx 3.65 \times 10^5$, $\delta \approx -0.137$, $\gamma \approx -0.369$. $D_f$ represents the tokens used for MLM-CLM, and $N$ is the number of parameters, with $k$, $\delta$, and $\gamma$ as fitting coefficients. For instance, a $10\times$ increase in $D_f$ would roughly triple the model size and double $D_t$. We validate these findings by evaluating the compute ratio of CLM pre-training under four specified parameters and FLOPs, as shown in Figure A14 (left), finding that MLM-CLM generally outperforms MLM-S. Specifically, $D_t/(D_t + D_f)$ ranges from 10% to 20% of the compute budget for CLM pre-training. Figure A14 (right) schematically illustrates the learning curves of two 85M (3e19 FLOPs) models, with MLM-CLM achieving similar or better loss levels with equal or fewer tokens.

## L. MoE Scaling

We find that the scaling behaviors of sparse parameter counts in Mixture of Expert (MoE) models are remarkably similar to those of dense model sizes, potentially allowing for a reduced compute budget for modeling scaling behaviors due to less activated parameters per token.

In our experiments, we evaluate MoE models ranging from

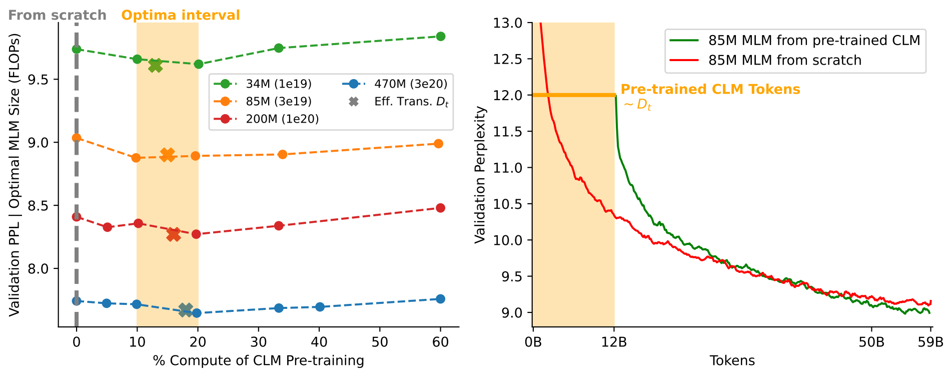

*Figure A14.* **Left**: Valid perplexity of % compute allocated for the CLM pre-training. For instance, % compute indicates first training on CLM and then the rest compute fine-tuning on MLM. The optimal CLM pre-training % compute range with [10, 20]. And the fitted $D_t/(D_t + D_f)$ drops in the optimal loss range. **Right**: Comparison of validation perplexity for models trained from scratch (red) and those fine-tuned from a pre-trained CLM (green), demonstrating that fine-tuning from a CLM reduces perplexity with similar or even fewer tokens.

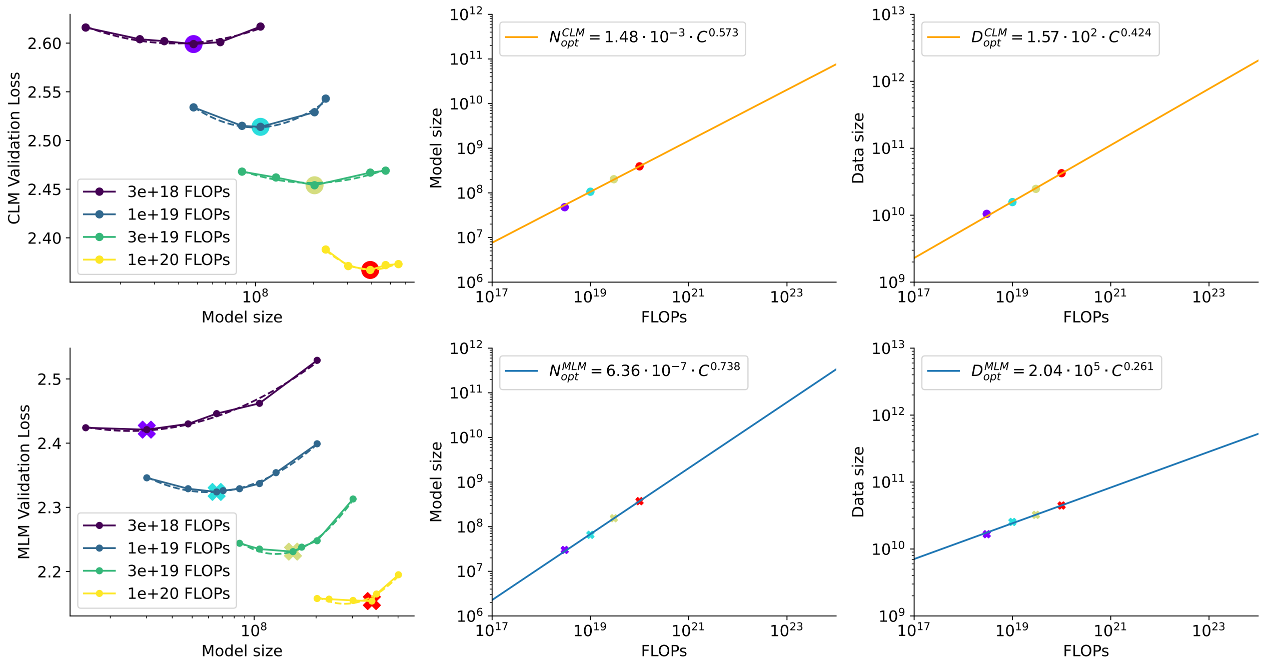

*Figure A15.* **Scaling laws of MoE.**The scaling behaviors of sparse parameter counts (8 experts) in MoE models, highlighting IsoFLOPs curves for different model sizes and FLOPs configurations. Each graph represents the relationship between model size, FLOPs, and validation loss for both CLM and MLM using MoE configurations. The power-law fits indicate optimal model size and data requirements for efficient scaling, showing that MoE models closely align with dense models in terms of scaling efficiency, with power-law coefficients for MoE-CLM and MoE-MLM approximating those of their dense counterparts. This suggests that MoE models can achieve similar scaling behaviors with potentially lower computational costs.

10M to 500M sparse parameter counts, using a model size of 17 with eight experts, following the settings outlined in Mixtral of experts (Jiang et al., 2024), including its load-balancing scheme. The figure below shows different IsoFLOPs curves. Notably, the FLOPs here are calculated based on sparse parameters rather than actually activated ones. We use the method described in the main text to select optimal loss points and fit these around the sample points, enabling us to project the optimal model size and number of tokens for larger models (center and right). We observe that the power-law coefficients for CLM and MLM are similar to those of dense models, with MoE CLM vs. Dense CLM at approximately 0.57 vs. 0.58, and MoE MLM vs. Dense MLM at 0.74 vs. 0.77.

Our study strictly focuses on models with eight experts, which may not be entirely rigorous. Clark et al. (Clark et al., 2022) proposed a unified scaling law defining effective training parameters for MoE, aiming to harmonize the scaling laws for Dense and MoE models. Investigation of biological data will be considered as future work.

## M. Combined Power-law

We applied the fitting function proposed by Chinchilla (Hoffmann et al., 2022), detailed in Equation 5, to model the effects of various factors on model performance. It can provide a loss prediction where neither the parameters or model size are not optimal allocation. This loss function simultaneously depends on parameters $N$ and $D$:

$$L(N, D) = \frac{A}{N^\alpha} + \frac{B}{D^\beta} + E \tag{5}$$

where $E$ denotes the irreducible loss. Parameters $A$, $B$, $\alpha$, and $\beta$ are learned through the fitting process. As $N \to \infty$ or $D \to \infty$, the function degenerates to a form similar to Equation 2, which indicates that it models the scenarios under perfect conditions of other variables.

Given that most of our training tokens are used for less than or equal to one epoch, and that the model size is prone to underfitting at fixed FLOPs, the asymptotic behaviors $L(N)$ at $D \to \infty$ and $L(D)$ at $N \to \infty$ are enough for determining the parameters in $L(N, D)$.

To enrich data points, we randomly added several FLOP counts into 25% of the model size and trained these models for 0.25, 0.5, 0.75, and 1 epoch. And we adopt the Huber loss to fit these coefficients: where LSE represents the log-sum-exp operator, and $\delta = 10^{-3}$. The terms $N_i$, $D_i$, and $L_i$ denote the model size, dataset size, and loss of the $i$-th run, respectively. We fitted the MLM validation loss from 110 samples and the CLM validation loss from 149 samples using grid search with $\alpha \in \{0, 0.5, \ldots, 2\}$, $\beta \in \{0, 0.5, \ldots, 2\}$, $e \in \{-1, -0.5, \ldots, 1\}$, $a \in \{0, 5, \ldots, 25\}$, and $b \in \{0, 5, \ldots, 25\}$. The final initialized parameters of CLM and MLM both are $[e, a, b, \alpha, \beta] = [1, 5, 10, 0.5, 0.5]$. We set the maximum number of iterations to 1000, and the two objectives were essentially achieved after 360 iterations. The exponential powers of learned $a$ and $b$ yielded the coefficients $A$, $B$, which were reported in Table A8.

Substituting all learned coefficients into the following Equation from the original Chinchilla paper:

$$N_{\text{opt}}(C) = G\left(\frac{C}{6}\right)^a, \quad D_{\text{opt}}(C) = G^{-1}\left(\frac{C}{6}\right)^b$$
$$where \quad G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}, \quad a = \frac{\beta}{\alpha+\beta}, \quad b = \frac{\alpha}{\alpha+\beta}. \tag{7}$$

The results closely approximate the trends given in Equations 1 and 2, confirming our overall findings.

## N. IsoLoss

In addition to using the seven different FLOPs counts reported in the main text to determine the optimal model sizes and fit our scaling law, we also incorporated additional model points into our analysis. We trained using the final loss points of all the CLM and MLM that are run. Figure A16 depicts the contour of the fitted function $L$ and the efficient frontier as a red dashed line, presented in log-log space. The frontier interval of Figure 2 is computed from this observation. From this approach, it revealed the scaling exponents for model size to be 0.77 in MLM and 0.57 in CLM, very similar to the IsoFLOPs profiling method in Section 3.

## O. Scaling law for training two models

When our goal is to optimize both CLM and MLM simultaneously, the strategic allocation of compute resources between these two objectives becomes essential. To facilitate this, we equalize model parameters across objectives to assess specific compute budgets for dual-objective training. Specifically, we seek the compute budgets, $C_{\text{MLM}}$ and $C_{\text{CLM}}$, for configurations where the optimal model size is the same, i.e., $N(C_{\text{MLM}}) = N(C_{\text{CLM}})$. These individual computations are then aggregated to formulate the overall compute budget:

$$C_{\text{sum}}(N) = C_{\text{MLM}}(N) + C_{\text{CLM}}(N) \tag{8}$$
$$= \left(\frac{6.2 \times 10^{-8}}{N}\right)^{0.776} + \left(\frac{1.25 \times 10^{-3}}{N}\right)^{0.578} \tag{9}$$

These two objectives share the same parameter size, their compute budget $C$ and the number of training tokens $D$ differ. Thus we further introduce a model-to-ratio $r(N)$ as $D_{\text{MLM}}(N)/D_{\text{CLM}}(N)$. We then achieve the relationship between $N$ and $C_{\text{sum}}$ by a fitted power-law (Figure A17) form:

$$\begin{cases} N(C_{\text{sum}}) \approx 1.497 \times 10^{-6} \times C_{\text{sum}}^{0.703} \\ r(N) \approx 8.449 \times 10^4 \times N^{-0.392} \end{cases} \tag{10}$$

$$\min_{a,b,e,\alpha,\beta} \sum_{\text{i}} \text{Huber}_\delta \left(\text{LSE}\left(a - \alpha \log N_i, b - \beta \log D_i, e - \log L_i\right)\right), \tag{6}$$

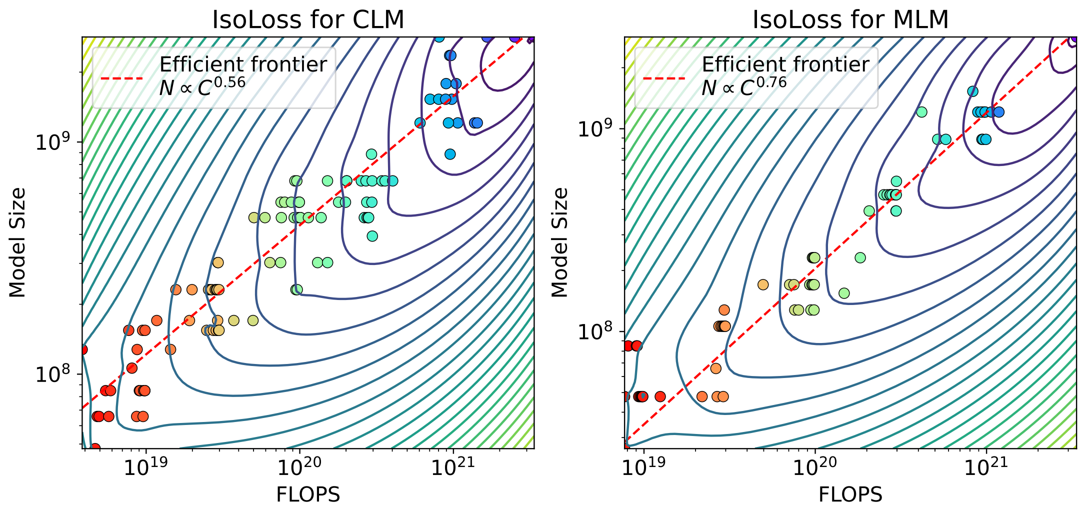

*Figure A16.* **Parametric fit for CLM and MLM**. Unlike the IsoFLOPs method used in the main text to select the optimal model size, these plots use all available data points to fit the models. The left panel shows the contour of the function $L$ and the efficient frontier (indicated by the red dashed line) for the CLM, and the right panel for the MLM. The rainbow dots represent identical loss. The results closely align with using the IsoFLOPs profiling method.

*Table A8.* Coefficient of Equation 5

| Objective | $A$ | $B$ | $\alpha$ | $\beta$ |
|---|---|---|---|---|
| CLM | 143.9 | 22036.5 | 0.367 | 0.496 |
| MLM | 3.365 | 7.569 | 0.042 | 0.099 |

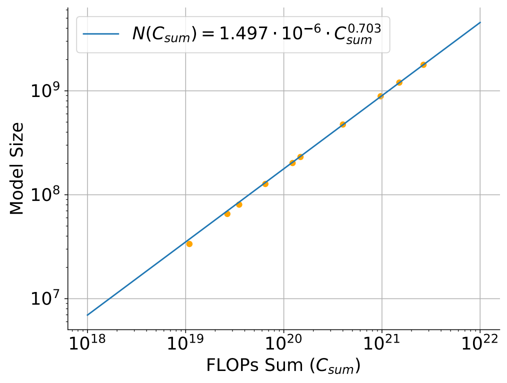

*Figure A17.* Compute allocation for two objectives with the same model size.

The ratio $r(N)$ informs us about the allocation proportion of training tokens. Specifically, under equal parameters, the data for MLM should exceed that for CLM until a 10B threshold (achieving a 1:1) is reached, after which more training tokens are allocated to CLM.

## P. Training Procedure

We conducted all experiments using Ampere A100 GPUs (80G) equipped with NVLink, utilizing the GLM framework (Zeng et al., 2022; Du et al., 2022) developed based on DeepSpeed and Megatron. Our approach predominantly utilized data parallelism, avoiding model parallelism and pipeline parallelism to simplify deployment. Modifications were made to the standard Transformer architecture (Vaswani et al., 2017), adopting a DeepNorm (Wang et al., 2024) strategy and layer normalization (Ba et al., 2016). The activation function was set to GLU (Shazeer, 2020), RoPE (Su et al., 2024) was used to encode position, similar to the settings found in the Transformer++ architecture (Touvron et al., 2023a). We further adopt FlashAttention (Dao et al., 2022) to accelerate our training process. The used max LR empirically found to range between $6 \times 10^{-4}$ and $1.2 \times 10^{-4}$ from small to large model size, was used along with a cosine decay strategy to reduce it to $0.1 \times$ max LR. Both CLM and MLM were trained under similar settings for model size, with a consistent LR and a minimum warm-up period of 2.5% steps, extending to at least 100K training steps. All sequences were set to a length of 1024, with sequences concatenated using an `<EOS>` delimiter. Based on findings related to loss magnitude and batch size (McCandlish et al., 2018). The AdamW optimizer (Loshchilov & Hutter, 2017) was used with $\beta_1 = 0.9$, $\beta_2 = 0.95, \epsilon = 1 \times 10^{-8}$, and a weight decay of $0.01$. All experiments omitted the dropout (it reduced the capacity to hinder model scaling) and trained with bfloat16. Most pre-training experiments were confined to the $\leq 1$ epoch, with some models extending up to 30% beyond one epoch. For the transfer learning setting, we load the finished checkpoint of the pre-training model and disregard the pre-trained optimized state, and learn rest tokens with warmup 5% steps the max LR.

## Q. Model Parameters

Table A9 details the sizes and configurations of all models utilized in this research, training only with data parallel expcept 10B with tensor parallel size 2:

*Table A9.* **All model hyperparameters.** Several of the models presented have been trained using various learning rate schedules and differing amounts of training tokens.

| params | d_model | ffw | kv_size | head_num | layers |
|---|---|---|---|---|---|
| 4M | 192 | 512 | 24 | 8 | 8 |
| 5M | 256 | 683 | 32 | 8 | 7 |
| 6M | 256 | 683 | 32 | 8 | 8 |
| 10M | 320 | 853 | 40 | 8 | 8 |
| 13M | 320 | 1280 | 40 | 8 | 8 |
| 19M | 448 | 1194 | 64 | 7 | 8 |
| 25M | 512 | 1365 | 64 | 8 | 8 |
| 34M | 512 | 2048 | 64 | 8 | 8 |
| 40M | 576 | 1536 | 64 | 8 | 10 |
| 47M | 576 | 1536 | 64 | 9 | 12 |
| 66M | 640 | 2560 | 64 | 10 | 10 |
| 77M | 480 | 1280 | 24 | 20 | 28 |
| 85M | 768 | 2048 | 64 | 12 | 12 |
| 106M | 768 | 2048 | 48 | 16 | 15 |
| 127M | 768 | 2048 | 48 | 16 | 18 |
| 154M | 896 | 2389 | 64 | 14 | 16 |
| 157M | 640 | 1707 | 32 | 20 | 32 |
| 170M | 768 | 2048 | 48 | 16 | 24 |
| 200M | 896 | 2389 | 64 | 14 | 21 |
| 230M | 896 | 2389 | 64 | 14 | 24 |
| 300M | 1024 | 2731 | 64 | 16 | 24 |
| 393M | 1280 | 3413 | 80 | 16 | 20 |
| 470M | 1280 | 3413 | 80 | 16 | 24 |
| 550M | 1280 | 3413 | 80 | 16 | 28 |
| 670M | 1536 | 4096 | 96 | 16 | 24 |
| 880M | 1792 | 4778 | 64 | 28 | 23 |
| 1.2B | 2048 | 5461 | 64 | 32 | 24 |
| 1.5B | 2304 | 6144 | 64 | 36 | 24 |
| 1.7B | 2304 | 6144 | 64 | 36 | 28 |
| 2.0B | 2560 | 6832 | 64 | 40 | 26 |
| 2.4B | 2560 | 6832 | 64 | 40 | 30 |
| 2.8B | 2560 | 6832 | 64 | 40 | 36 |
| 3.1B | 2688 | 7168 | 64 | 42 | 36 |
| 3.4B | 2816 | 15040 | 128 | 22 | 22 |
| 4.0B | 3072 | 8192 | 128 | 24 | 36 |
| 5.7B | 3328 | 8874 | 128 | 26 | 40 |
| 6.2B | 3584 | 9556 | 128 | 28 | 40 |
| 7.2B | 4096 | 10923 | 128 | 36 | 36 |
| 10.7B | 4352 | 11605 | 136 | 32 | 47 |