Protein Sequence Domain Annotation using Language Models

Arpan Sarkar* **Kumaresh Krishnan*** **Sean R. Eddy**
Harvard University
arpan_sarkar@g.harvard.edu
{kumaresh_krishnan, seaneddy}@fas.harvard.edu

Abstract

Protein function inference relies on annotating protein domains via sequence similarity, often modeled through profile Hidden Markov Models (profile HMMs), which capture evolutionary diversity within related domains. However, profile HMMs make strong simplifying independence assumptions when modeling residues in a sequence. Here, we introduce PSALM (Protein Sequence Annotation using Language Models), a hierarchical approach that relaxes these assumptions and uses representations of protein sequences learned by protein language models to enable high-sensitivity, high-specificity residue-level protein sequence annotation. We also develop a benchmark for protein sequence domain annotation, where training and test sequences have been rigorously split to share no similarity between any of their domains at a given threshold of sequence identity. Prior benchmarks, which split one domain family at a time, do not support methods for annotating multidomain proteins, where training and test sequences need to have multiple domains from different families. We validate PSALM's performance on this benchmark and highlight PSALM as a promising alternative to HMMER, a state-of-the-art profile HMM-based method, for protein sequence annotation.

1 Introduction

Proteins consist of structural and functional units called domains, which are conserved through evolution. The primary aim of protein sequence annotation is to identify and characterize these domains, as understanding their individual functions can provide insight into the protein's overall biological role. Given the challenges of experimental protein function characterization, function is often inferred through sequence similarity (homology) to domains with known roles [1, 2]. As protein sequence databases and the number of unannotated sequences expand rapidly [3], efficient annotation methods have become vital for leveraging this growing resource to understand molecular biology and evolution.

The state of the art uses profile hidden Markov models (profile HMMs) to detect domains [4] and profile/profile comparison to identify homologous domains [5]. Databases like Pfam [6] organize millions of protein sequences into approximately 20,000 domain families, each represented by a profile HMM. Annotation is achieved by comparing a protein sequence against these domain profiles, rather than scanning millions of individual sequences. This approach not only classifies the full protein but also identifies the composition and boundaries of each domain at the residue level, both enabling functional inference and preventing the "transitive identification catastrophe," where function is mistakenly transferred between sequences due to homology of unrelated domains [7].

*These authors have equal contribution

Machine Learning for Structural Biology Workshop, NeurIPS 2024.

In this work, we introduce Protein Sequence Annotation with Language Models (PSALM) and show that ESM-2, a pre-trained general-purpose protein language model (pLM) [8], can be extended to predict *residue-level* sequence annotations. Our contributions:

- To the best of our knowledge, PSALM is the **first deep learning approach for residue-level protein sequence domain annotation**. While deep learning methods have been applied extensively to protein sequence annotation, most attempts focus on recognition/classification of entire protein sequences [9–13], rather than recognition of individual domain subsequences in a longer target sequence, which is a different problem.
- We demonstrate that, by utilizing pLMs, we can relax the strong, simplifying independence assumptions made by profile HMMs when modeling residues in a protein sequence and can predict *residue-level* sequence annotations with **greater sensitivity and specificity than state-of-the-art profile HMM-based methods**. Predicted homology from PSALM accurately annotates domain boundaries, multi-domain proteins, conserved domains across distantly-related protein sequences, and even domains that are currently unannotated.
- We develop a **benchmark for protein sequence domain annotation**, where training and test sequences have been rigorously split to share no similarity between any of their domains at a given threshold of sequence identity, enabling challenging, realistic evaluation of domain annotation methods across a wide range of domain families and sequence similarity. Prior benchmarks, which split one domain family at a time, do not support methods for annotating multidomain proteins, where training and test sequences need to have multiple domains from different families.

2 Related work

Profile HMMs This approach uses curated multiple sequence alignments (MSAs) of related domains, which reveal patterns of conservation and variability at the residue level, to model consensus using "match", "insert", and "delete" hidden states [2, 14]. These models serve as templates for comparison against the sequence of interest, enabling the identification of domains by finding subsequences that match the profile HMMs. Sequences with multiple, unrelated domains will require the use of multiple profile HMMs for annotation. HMMER [4] is the state-of-the-art protein sequence domain annotation method and underlies many different databases, which organize related domains into MSAs and profile HMMs at varying levels of granularity, enabling profile-based annotations at the superfamily [15], family [6], and sub-family [16] levels. While profile HMMs have enabled sensitive, high-coverage, large-scale annotation of the known protein universe [6], they make two limiting assumptions. Profile HMMs assume both that the observed residues are conditionally independent given the hidden state and that the transition to the next hidden state depends only on the current state (Markov property). While these assumptions simplify the modeling process, they may prevent profile HMMs from capturing complex dependencies between residues in a sequence by ignoring how residues distant in the one-dimensional sequence can interact with each other, especially over long distances, in the folded three-dimensional structure of the protein.

Deep Models Recent breakthroughs like AlphaFold2 [17] prove that deep-learning-based approaches can learn these complex relationships from sequence data. However, efforts to apply deep learning methods to predict protein function from sequence have either focused on predicting ontology-based functional annotation at the sequence level [18–21] or recognizing homology at the sequence level [10–13]. To our knowledge, ProtENN, an ensemble of convolutional neural networks, represents the first attempt to predict Pfam domains directly from protein sequences [9]. ProtENN, however, is constrained to make one domain prediction per input sequence and cannot natively identify domain boundaries or multiple domains within a sequence without ad hoc post-processing. Additionally, ProtENN cannot provide information on the contribution of an individual residue to a predicted annotation.

3 PSALM overview

We frame the residue-level protein sequence annotation problem as a mapping from a protein sequence $\mathbf{x} = (x_1, x_2, \ldots, x_L)$ to a sequence of domain family labels $\mathbf{y} = (y_1, y_2, \ldots, y_L)$. Each residue x_i represents one of 25 possible amino acids (20 canonical, 2 non-canonical, and 3 ambiguous amino

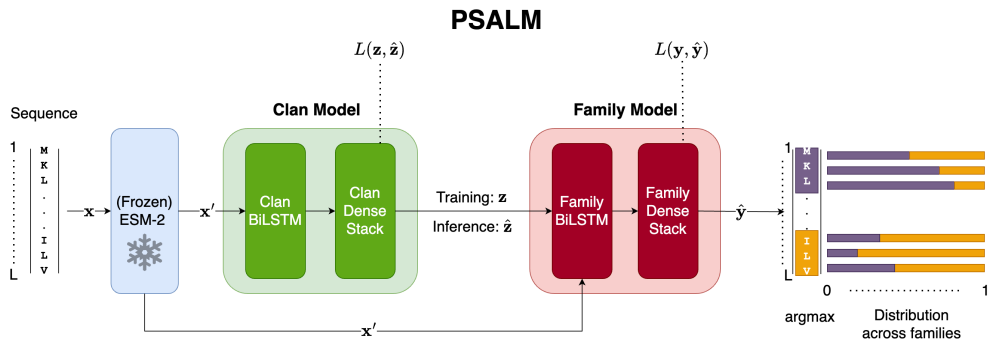

Figure 1: **Overview of residue-level protein sequence annotation with PSALM.** A sequence $\mathbf{x}$ of length L is embedded as $\mathbf{x}'$ with a frozen ESM-2. The PSALM clan and family models predict the clan annotations $\hat{\mathbf{z}}$ and family annotations $\hat{\mathbf{y}}$, respectively, and are trained to minimize cross-entropy loss $L(\cdot)$. During training only, the true clan annotations $\mathbf{z}$ are provided to the family model. Here, the example outputs are predicted across a set of 2 families.

acid characters), while each label y_i corresponds to one of D domain families, with $D + 1$ for none. Approximately 23% of protein sequences and 47% of residues in UniProt lack a Pfam domain annotation [6]. The goal is to train a model to predict the domain family for each residue:

$$\hat{y}_i = \arg\max_f P(Y_i = f|\mathbf{x}), \tag{1}$$

where $P(Y_i)$ is the probability distribution over the $D + 1$ domain family labels.

To numerically encode protein sequences for machine learning tasks, we use a pre-trained protein language model (pLM). Specifically, we employ ESM-2 650M, which generates residue-level sequence embeddings $\mathbf{x}'$ by capturing long-range interactions and contextual information [8]. These embeddings $\mathbf{x}'$ replace the original sequence $\mathbf{x}$ for downstream tasks. In this approach, we keep the ESM-2 weights frozen.

PSALM uses a hierarchical approach (Fig. 1) that considers both individual protein domain families and clans, which are collections of evolutionarily related (homologous) protein domain families categorized by Pfam [22]. In Pfam 35.0, approximately 45% of the 19,632 Pfam families are grouped into 655 clans, and a family can only belong to at most one clan. While the main task is to predict the domain family at each residue, clan-level predictions provide an interpretable intermediate step, aiding functional and structural insights when family-level information is unclear. For each residue, PSALM predicts the clan ($\hat{z}_i$) and family ($\hat{y}_i$) as follows:

$$\hat{z}_i = \arg\max_c P(Z_i = c|\mathbf{x}') \tag{2}$$

$$\hat{y}_i = \arg\max_f P(Y_i = f|Z_i = C(f), \mathbf{x}')P(Z_i = C(f)|\mathbf{x}'), \tag{3}$$

where $C(f)$ maps a family to its corresponding clan. By jointly modeling clans and families, PSALM enhances interpretability and accuracy, ensuring robust residue-level domain annotations. Additional details, including those concerning choice of model architecture, training, and hyperparameter selection, are provided in Appendix A.2)

4 Benchmarking

Data In many machine learning contexts, data samples are assumed independent, allowing for random training-test splits. However, protein sequences share evolutionary relationships, making random splits unreliable and likely to inflate performance estimates [23–27]. To address this, we used Pfam-A Seed 35.0, a dataset of approximately 1.2 million proteins from 20,000 domain families [6], and split it into a training set of 517,936 sequences and five test subsets, from across which a validation set was sampled, resulting in a total of 73,286 test sequences and 5,775 validation sequences. Each test subset was defined by the maximum percent identity (PID) that any domain

Table 1: Residue-level domain annotation benchmark on Pfam Seed dataset

PID	Model	Clan TPR	Clan FPR	Clan F1	Clan MCC	Family TPR	Family FPR	Family F1	Family MCC
0-20%	HMMER*	0.694	0.033	0.819	0.642	0.659	0.033	0.810	0.636
	$PSALM_{650}$	**0.944**	**0.022**	**0.985**	**0.957**	**0.750**	**0.012**	**0.978**	**0.947**
	$PSALM_{OH}$	0.490	0.100	0.764	0.559	0.089	0.022	0.236	0.203
20-40%	HMMER*	0.907	0.043	0.941	0.862	**0.876**	0.043	0.939	0.861
	$PSALM_{650}$	**0.966**	**0.020**	**0.985**	**0.964**	0.845	**0.015**	**0.982**	**0.959**
	$PSALM_{OH}$	0.516	0.107	0.780	0.602	0.102	0.023	0.282	0.251
40-60%	HMMER*	0.951	0.058	0.957	0.898	0.921	0.058	0.956	0.896
	$PSALM_{650}$	**0.977**	**0.020**	**0.986**	**0.966**	**0.924**	**0.017**	**0.984**	**0.964**
	$PSALM_{OH}$	0.666	0.104	0.835	0.671	0.159	0.029	0.430	0.363
60-80%	HMMER*	0.974	0.059	0.971	0.924	0.946	0.059	0.970	0.923
	$PSALM_{650}$	**0.984**	**0.018**	**0.988**	**0.970**	**0.957**	**0.016**	**0.988**	**0.968**
	$PSALM_{OH}$	0.788	0.094	0.890	0.745	0.216	0.027	0.573	0.478
80-100%	HMMER*	0.977	0.051	0.972	0.935	0.950	0.051	0.971	0.934
	$PSALM_{650}$	**0.981**	**0.015**	**0.986**	**0.969**	**0.967**	**0.012**	**0.986**	**0.968**
	$PSALM_{OH}$	0.877	0.066	0.925	0.836	0.282	0.018	0.709	0.630

in any test sequence shares with any domain in any sequence in the the training set. This ensures varying levels of similarity, from $0 <$ max PID ≤ 20 to $80 <$ max PID ≤ 100. Further details on benchmark dataset creation are provided in Appendex A.1.

Baselines We establish two baseline methods for comparison. We use HMMER, the current state-of-the-art protein sequence annotation method, to build profile HMMs from MSAs of the ground truth domains in the training set, denoted as HMMER*, and use these profiles to annotate the test sequences with `hmmscan`. This allows us to evaluate how a state-of-the-art profile HMM method compares to PSALM when using the same training and testing sets. Additionally, we implement a variant of PSALM, denoted as $PSALM_{OH}$, where one-hot embeddings for each amino acid in a protein sequence are utilized instead of embeddings from the pre-trained protein language model ESM-2. This comparison helps discern whether differences in performance between PSALM and HMMER* are influenced by ESM-2 or solely by the subsequent neural network architecture. Additionally, we assess both the capacity of PSALM by evaluating performance across different ESM-2 model sizes (Appendix A.3) and the impact of intermediate clan predictions by evaluating a "family-only" PSALM (Appendix A.4). The PSALM model presented in the main results in Table 1, which uses a frozen ESM-2 650M and has intermediate clan predictions, is denoted as $PSALM_{650}$.

Evaluation Protein sequence databases have vastly more negatives than positives, requiring extremely low (essentially zero) and controllable false positive rate (FPR), as false annotations are amplified and propagated to additional sequences by later searches. Methods in this field are typically benchmarked for the sensitivity or true positive rate (TPR) they can achieve at a high specificity (low FPR). We also report the F1 score and Matthews Correlation Coefficient (MCC). Here, FPR is defined as the fraction of true negative residues (shuffled, preserving residue composition) incorrectly identified as homologous to a Pfam protein domain family, and TPR is defined as the fraction of residues in ground truth domains correctly identified.

We highlight the key observations from the sequence annotation benchmark. $PSALM_{650}$ demonstrates superior performance in domain annotation, accurately annotating a substantial portion of true domain regions while consistently calling fewer false positives compared to HMMER* (Table 1), with the single exception being family TPR at the 20-40 PID range test subset. PSALM performance strictly increases with ESM-2 model size (Appendix A.3) and the inclusion of intermediate clan predictions (Appendix A.4). The one-hot baseline $PSALM_{OH}$ learns to predict clan annotations but consistently has the lowest TPR for family annotation.

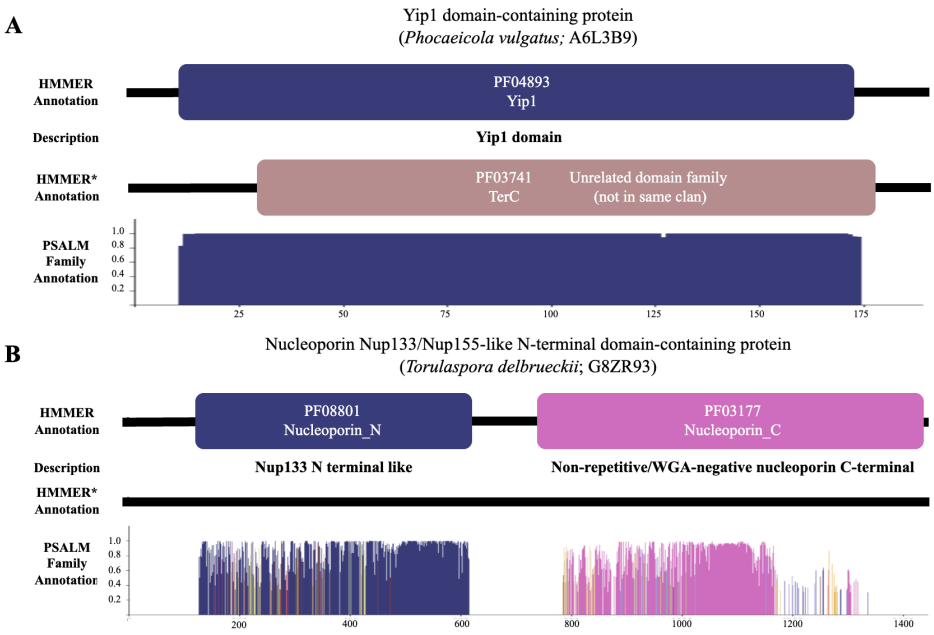

Figure 2: Comparison of PSALM and HMMER* annotations to the ground truth HMMER annotations for two selected protein sequences from the 0-20% PID test subset.

5 Examples

In Figure 2, we compare the PSALM_{650} and HMMER* annotations to the ground truth annotations determined by HMMER, focusing on two protein sequences drawn from the 0-20% PID test subset, which contains the domains most distantly related to those in the training set. PSALM is able to identify domains in this test subset that HMMER* either misidentifies (Fig. 2 A) or fails to identify (Fig. 2 B). Figure 2 B further demonstrates both PSALM's ability to identify multiple domains in a single input sequence and PSALM's inability to annotate the full C-terminal domain, which is too diverged from related domains in the training set for PSALM to fully identify.

6 Conclusions and limitations

We introduce PSALM, a highly sensitive and specific protein sequence annotation method, which extends the capabilities of self-supervised pLMs with just a few hundred thousand protein sequences, enabling interpretable residue-level annotations at both the clan and family levels. Code, data, and models are available in the `protein-sequence-annotation` Python package and `Protein-Sequence-Annotation/PSALM` GitHub repository. We address the current limitations and potential future research directions of this approach with the following points.

Data leakage Despite our efforts to mitigate it, information from the test set may still contribute to training through the millions of sequences used to train ESM-2. While we exclude sequences used to train ESM-2 from our 0-20 PID test subset, indirect leakage through homology remains a possibility. Ideally, retraining ESM-2 from scratch on our training data would provide better insight into the out-of-distribution generalization capabilities of PSALM. We have not done this in the present work because of the compute demand for training a pLM like ESM-2, but we plan to rigorously split a much larger set of proteins to train a pLM from scratch.

Domain calling PSALM cannot distinguish between repeated domains occurring consecutively or accurately resolve split domains. For example, if a domain repeats immediately after itself, PSALM labels the entire two domain block instead of recognizing two separate domains within it. Similarly, when a domain is split, PSALM identifies the two halves as separate domains from the same family, rather than as originating from a single domain. We aim to address this by explicitly modeling the domain boundaries and developing domain-calling algorithms.

References

[1] William R Pearson. An Introduction to Sequence Similarity ("Homology") Searching. *Current Protocols in Bioinformatics*, 42(1):3–1, 2013.

[2] Sean R. Eddy. Profile hidden Markov models. *Bioinformatics (Oxford, England)*, 14(9): 755–763, 1998.

[3] UniProt Consortium. UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Research*, 51(D1):D523–D531, 2023.

[4] Sean R Eddy. Accelerated Profile HMM Searches. *PLoS Computational Biology*, 7(10): e1002195, 2011.

[5] Michael Remmert, Andreas Biegert, Andreas Hauser, and Johannes Söding. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods*, 9(2): 173–175, 2012.

[6] Jaina Mistry, Sara Chuguransky, Lowri Williams, Matloob Qureshi, Gustavo A Salazar, Erik LL Sonnhammer, Silvio CE Tosatto, Lisanna Paladin, Shriya Raj, Lorna J Richardson, et al. Pfam: The protein families database in 2021. *Nucleic Acids Research*, 49(D1):D412–D419, 2021.

[7] Tobias Doerks, Amos Bairoch, and Peer Bork. Protein annotation: detective work for function prediction. *Trends in Genetics*, 14(6):248–250, 1998.

[8] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[9] Maxwell L Bileschi, David Belanger, Drew H Bryant, Theo Sanderson, Brandon Carter, D Sculley, Alex Bateman, Mark A DePristo, and Lucy J Colwell. Using deep learning to annotate the protein universe. *Nature Biotechnology*, 40(6):932–937, 2022.

[10] Michael Heinzinger, Maria Littmann, Ian Sillitoe, Nicola Bordin, Christine Orengo, and Burkhard Rost. Contrastive learning on protein embeddings enlightens midnight zone. *NAR Genomics and Bioinformatics*, 4(2):lqac043, 2022.

[11] Vamsi Nallapareddy, Nicola Bordin, Ian Sillitoe, Michael Heinzinger, Maria Littmann, Vaishali P Waman, Neeladri Sen, Burkhard Rost, and Christine Orengo. CATHe: detection of remote homologues for CATH superfamilies using embeddings from protein language models. *Bioinformatics*, 39(1):btad029, 2023.

[12] Kamil Kaminski, Jan Ludwiczak, Kamil Pawlicki, Vikram Alva, and Stanislaw Dunin-Horkawicz. pLM-BLAST: distant homology detection based on direct comparison of sequence representations from protein language models. *Bioinformatics*, 39(10):btad579, 2023.

[13] Tymor Hamamsy, James T Morton, Robert Blackwell, Daniel Berenberg, Nicholas Carriero, Vladimir Gligorijevic, Charlie EM Strauss, Julia Koehler Leman, Kyunghyun Cho, and Richard Bonneau. Protein remote homology detection and structural alignment using deep learning. *Nature Biotechnology*, pages 1–11, 2023.

[14] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[15] Arun Prasad Pandurangan, Jonathan Stahlhacke, Matt E Oates, Ben Smithers, and Julian Gough. The SUPERFAMILY 2.0 database: a significant proteome update and a new webserver. *Nucleic Acids Research*, 47(D1):D490–D494, 2019.

[16] Paul D Thomas, Dustin Ebert, Anushya Muruganujan, Tremayne Mushayahama, Laurent-Philippe Albou, and Huaiyu Mi. PANTHER: Making genome-scale phylogenetics accessible to all. *Protein Science*, 31(1):8–22, 2022.

[17] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

[18] Yue Cao and Yang Shen. TALE: Transformer-based protein function Annotation with joint sequence–Label Embedding. *Bioinformatics*, 37(18):2825–2833, 2021.

[19] Jiajun Hong, Yongchao Luo, Yang Zhang, Junbiao Ying, Weiwei Xue, Tian Xie, Lin Tao, and Feng Zhu. Protein functional annotation of simultaneously improved stability, accuracy and false discovery rate achieved by a sequence-based deep learning. *Briefings in Bioinformatics*, 21(4):1437–1447, 2020.

[20] Maxat Kulmanov and Robert Hoehndorf. DeepGOPlus: improved protein function prediction from sequence. *Bioinformatics*, 36(2):422–429, 2020.

[21] Theo Sanderson, Maxwell L Bileschi, David Belanger, and Lucy J Colwell. ProteInfer, deep neural networks for protein functional inference. *elife*, 12:e80942, 2023.

[22] Robert D Finn, Jaina Mistry, Benjamin Schuster-Böckler, Sam Griffiths-Jones, Volker Hollich, Timo Lassmann, Simon Moxon, Mhairi Marshall, Ajay Khanna, Richard Durbin, et al. Pfam: clans, web tools and services. *Nucleic Acids Research*, 34(suppl_1):D247–D251, 2006.

[23] Johannes Söding and Michael Remmert. Protein sequence comparison and fold recognition: progress and good-practice benchmarking. *Current Opinion in Structural Biology*, 21(3): 404–411, 2011.

[24] Ian Walsh, Gianluca Pollastri, and Silvio CE Tosatto. Correct machine learning on protein sequences: a peer-reviewing perspective. *Briefings in Bioinformatics*, 17(5):831–840, 2016.

[25] David T Jones. Setting the standards for machine learning in biology. *Nature Reviews Molecular Cell Biology*, 20(11):659–660, 2019.

[26] Ian Walsh, Dmytro Fishman, Dario Garcia-Gasulla, Tiina Titma, Gianluca Pollastri, Jennifer Harrow, Fotis E Psomopoulos, and Silvio CE Tosatto. DOME: recommendations for supervised machine learning validation in biology. *Nature Methods*, 18(10):1122–1127, 2021.

[27] Samantha Petti and Sean R Eddy. Constructing benchmark test sets for biological sequence analysis using independent set algorithms. *PLOS Computational Biology*, 18(3):e1009492, 2022.

[28] Chris Sander and Reinhard Schneider. Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins: Structure, Function, and Bioinformatics*, 9 (1):56–68, 1991.

[29] Burkhard Rost. Twilight zone of protein sequence alignments. *Protein Engineering*, 12(2): 85–94, 1999.

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

[31] William R Pearson. Flexible sequence similarity searching with the FASTA3 program package. *Bioinformatics Methods and Protocols*, pages 185–219, 1999.

[32] Samuel Karlin and Stephen F Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proceedings of the National Academy of Sciences*, 87(6):2264–2268, 1990.

[33] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997.

[34] T Wessels and Christian W Omlin. Refining hidden Markov models with recurrent neural networks. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 2, pages 271–276. IEEE, 2000.

[35] Achille Salaün, Yohan Petetin, and François Desbouvries. Comparing the modeling powers of RNN and HMM. In *2019 18th IEEE International Conference on Machine Learning and Applications*, pages 1496–1499. IEEE, 2019.

[36] Ronald J Williams and David Zipser. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280, 1989.

[37] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A Appendix

A.1 Benchmark creation

The goal of benchmark creation is performed in two phases. In the first phase, we filter *domains* by a strict sequence percent identity (PID; Appendix A.1.1) to remove similar domains and ensure that test domains are sufficiently dissimilar from training domains. In the second phase, we retrieve the *full-length* sequences corresponding to these domains and categorize them by their similarity to the training set. We begin by collecting the 1.2M "seed domains" from Pfam-A Seed 35.0, a set of curated, representative domains for each domain family that are used to build the 20K Pfam profile HMMs [6]. We apply BLUE [27], a graph-based sequence splitting algorithm, to partition the seed domains into preliminary training and test sets, defining an edge between two domains as their pairwise PID)Appendix A.1.1). We choose a PID threshold of 25% to split the seed domains as a meaningful cutoff to differentiate structurally and functionally distinct domains – protein pairs that share >25% identity indicate similar structure and function [28], and less than 1-% of protein pairs sharing <25% identity have similar structures [29]. This filtering step results in 560K training domains and 190K test domains. The remaining 450K seed domains were discarded due to sharing $> 25\%$ PID with both test and training sets.

In order to effectively assess the ability of PSALM to identify multiple domains in a sequence (as opposed to annotating a pre-determined region of interest), the benchmark needs to contain *full-length* protein sequences. We retrieve the full-length sequences corresponding to these representative training and test domains from UniProt release March 2021, a comprehensive database of 230M protein sequences. This results in 517K training sequences. From the test set, we eliminate duplicate sequences also present in the training set. All sequences across both training and test sets are annotated via the `hmmscan` tool from HMMER [4] with strict inclusion thresholds (E-value < 0.001, bitscore ≥ 30) in order to identify domain hits that constitute a "ground truth", with special care to nested, contiguous domains, which may escape typical processing methods (Appendix A.1.3). For training and test purposes, family and clan labels are only assigned to ground truth domains. We discard sequences in test that do not contain an annotated ground truth domain represented by at least 20 ground truth domains from the same family in the training set, resulting in 73K test sequences.

Simply splitting domains by PID is not sufficient to enforce the same gaurantees on full-length, multi-domain proteins, as the full-length sequences may contain additional domains beyond the seed domains. Thus, we further categorize each retrieved test sequence by the maximum PID that any of its domains shares with any domain in the training set as a conservative proxy for its distance from the training set (Appendix A.1.2). We partition the test set into five subsets based on this maximum PID (Table 2), and a total of 6K validation sequences are sampled uniformly across the test subsets. Such a partition may result in test sequences that, for example, may be placed in the $80 <$ PID ≤ 100 subset due to a single domain closely related to one in the training set, whereas the test sequence may have several other domains that share significantly lower PID with domains in the training set (this is why the average PID is near the lower bound of the max PID range for many of the test subsets in Table 2). The domain coverage, defined as the average percent of residues in a sequence that are labeled by Pfam domains, is similar across all test subsets.

We address the potential for unannotated domains to introduce data leakage across the training and test sets by shuffling all subsequences without family and clan labels in the test sequences to disrupt possible domain structures, preserving 0^{th} order residue-composition [1, 4]. Since PSALM may be sensitive enough to identify unannotated domains, it is trained with these regions shuffled, to mitigate penalties for "false positives" (with respect to the ground truth annotations). Another source

Table 2: Partition of the test set into 5 subsets

Test splits	0-20%	20-40%	40-60%	60-80%	80-100%	Train	Val
Sequences	4,087	37,319	17,756	8,570	5,864	517,936	5,775
Families	543	2,456	1,952	1,731	1,697	14,811	2,097
Clans	180	414	365	341	319	646	388
Coverage	65.31%	59.74%	58.66%	62.02%	56.71%	60.41%	58.79%
Average PID	18.35%	27.64%	42.45%	58.08%	80.01%	NA	45.08%

of data leakage may arise from the millions of representative sequences from the UniRef50 database release April 2021 [30] used to train ESM-2. We identify that none of the 4,087 sequences in the $0 <$ PID ≤ 20 test subset were present as representative sequences in this version of UniRef50. However, UniRef50 may contain close homologs to the sequences in this test subset.

A.1.1 BLUE

We use the BLUE algorithm [27] to split the 1.2M Pfam Seed domains into preliminary train and test sets with a PID threshold of 25%. The pairwise PID between two sequences $\mathbf{x}$ and $\mathbf{y}$ is defined as follows:

$$\text{PID}(\mathbf{x}, \mathbf{y}) = \frac{\text{\# aligned residues}}{\min(\ell(\mathbf{x}), \ell(\mathbf{y}))}, \tag{4}$$

where $\ell(\mathbf{x})$ and $\ell(\mathbf{y})$ represent the lengths of sequences $\mathbf{x}$ and $\mathbf{y}$, respectively. If two domains are in the same family, PID is directly calculated from their seed alignment. If two domains are not in the same family but are in the same clan, they are aligned using the `glsearch` tool from the FASTA3 software package [31], which performs a "global-local" alignment to account for possible large differences in sequence length. If two domains are not in the same clan, they are assumed to share $< 25\%$ PID.

A.1.2 Maximum PID calculation

Once the full length test sequences have been retrieved and subsequently filtered, we compute, for each test sequence, the maximum PID between any of its annotated domains and any annotated domain in train via Algorithm 1. Each test sequence is assigned to a single (out of five) test subset based on its maxmimum PID.

Algorithm 1 Percent identity splitting test set

Require: train sequences $\mathcal{D}^{tr}$, test sequences $\mathcal{D}^{te}$, Pfam family profile HMMs $\mathcal{F}$
  Initialize an empty dictionary-like structure record_max_pids
  for $f \in \mathcal{F}$ **do**
    train_domains $\leftarrow$ `hmmsearch` f against $\mathcal{D}^{tr}$
    test_domains $\leftarrow$ `hmmsearch` f against $\mathcal{D}^{te}$
    for (domain,sequence_id) $\in$ test_domains **do**
      MSA $\leftarrow$ `hmmalign` domain to train_domains with f
      domain_pids $\leftarrow$ `esl-alipid` MSA
      max_pid $\leftarrow \max$(domain_pids)
      if sequence_id not in record_max_pids **then**
        record_max_pids[sequence_id] $\leftarrow$ max_pid
      else if max_pid $>$ record_max_pids[sequence_id] **then**
        record_max_pids[sequence_id] $\leftarrow$ max_pid
      end if
    end for
  end for
  Assign each sequence in $\mathcal{D}^{te}$ to a test split based on max pid

The `esl-alipid` tool calculates PID for all pairs of sequences for a given MSA, and is part of the EASEL software package, which can be downloaded together with HMMER [4].

A.1.3 Ground truth annotation

We determine "ground truth" by annotating full length sequence with Pfam Seed profile HMMs using `hmmscan` with strict inclusion criteria (E-value < 0.001, bitscore ≥ 30). The highest-scoring annotation at each residue is taken as ground truth, but additional post-processing is necessary to ensure that "nested" domain structures are retained. This is accomplished by considering the "match strings" that HMMER generates for an alignment. The match strings contain characters that represent matches, where residues align to a given domain profile, and characters that represent inserts, where unaligned residues are inserted into the sequence relative to the domain profile. Annotating the

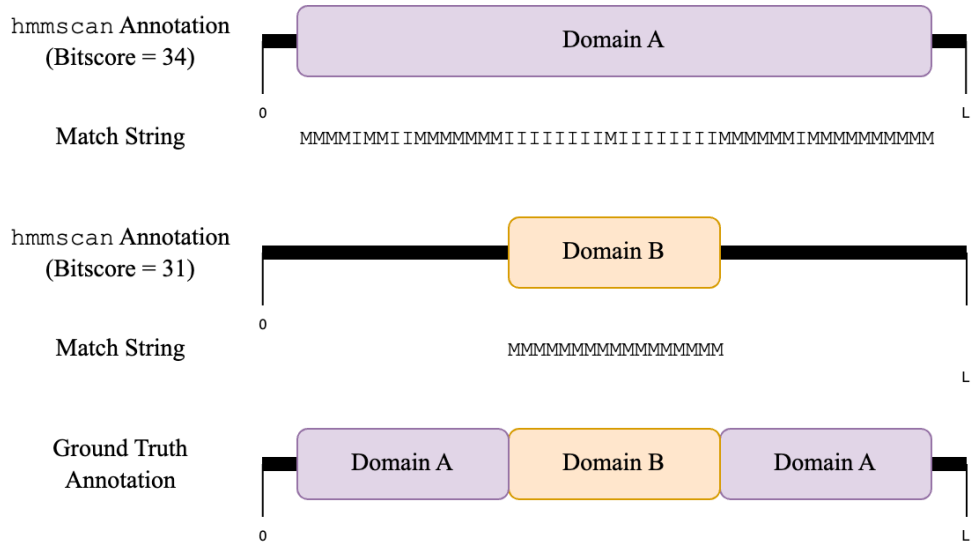

Figure 3: **A schematic of nested domains with two domains A and B in the nested format A-B-A.** As A is annotated with a higher score than B and overlaps with B, annotating residues only via highest score will fail to include domain B. Using the match state strings to identify smoothed maximal segments preserves the nested domain structure in the ground truth annotation.

highest-scoring match state at each residue preserves nested domain structure in the ground truth annotations (Fig. 3).

In a match string, regions with majority matches may contain a few inserts and vice versa. To prevent frequently alternating annotations in the ground truth, we smooth the match and insert states in the match string by identifying maximal scoring segments within the sequence. We assign insert states a positive score and match states a negative score. The segment of the sequence with the greatest aggregate score is known as the maximal segment [32], and all residues in the maximal segment are denoted as insert states. The scores s_i for each state are inferred from the match string for a given sequence:

$$s_i \propto \log\left(\frac{q_i}{p_i}\right), \tag{5}$$

where p_i is the frequency with which the state appears in the match string, and q_i is a state's target frequency, with $\sum_i p_i = 1$ and $\sum_i q_i = 1$. We set the insert state target frequency at 0.85, the match state target frequency at 0.15, and the length threshold at 20, below which maximal segments are ignored.

A.2 Implementation details

The clan and family models follow a similar structure and are trained separately. Protein sequences are initially embedded, providing a sequence of continuous, context-dependent residue-level embeddings as a replacement for the sequence of amino acid characters typically used as input to profile HMMs. The resulting embeddings are then passed into a single bidirectional Long Short-Term Memory (BiLSTM) layer to capture sequential dependencies [33] in the forwards and backwards directions. This bidirectional approach was chosen to mimic the backward pass in profile HMM optimization, aligning with their ability to account for sequence contexts in both directions [2]. RNNs, including BiLSTMs, have been shown to generalize profile HMMs by extending their capacity to model complex, nonlinear dependencies in sequential data, while retaining state-based representations [34, 35]. The choice of BiLSTM was made deliberately to introduce as few changes as possible, ensuring that the observed performance improvements could be attributed primarily to the use of the protein language model, rather than architectural differences from profile HMMs. The output from the BiLSTM layer is subsequently decoded using a stack of three dense layers, scaled to the number of clans or family labels, to produce logits across the prediction space. Probabilities are computed by applying softmax to the logits generated by each model.

Model	# Params		Learning Rate	
	Clan	Family	Clan	Family
PSALM_{650}	69M	166M	5e−4	5e−5
PSALM_{150}	18M	67M	5e−4	5e−5
PSALM_{35}	10M	47M	5e−4	5e−5
PSALM_{8}	5M	29M	5e−4	5e−5
PSALM_{OH}	56M	153M	1e−4	1e−5

Table 3: Number of parameters and learning rates (LR) for PSALM models.

Both PSALM and PSALM-onehot are trained using cross entropy loss over the entire sequence for both family and clan annotations. The family model is trained via teacher forcing, where it is provided the correct clan annotation for each residue in order to mitigate error propagation [36]. For training all PSALM+ESM-2 models, we use ADAM optimizer [37] with initial learning rate $5e-4$ for the clan model and $5e-5$ for the family model. These values were selected via hyperparameter tuning from across the following learning rates: $[1e-3, 5e-4, 1e-4, 5e-5, 1e-5]$. A similar hyperparameter search results in a learning rate of $1e-4$ for the PSALM_{OH} clan model and $1e-5$ for the family model. We employ a learning rate scheduler that reduces the learning rate by a factor of $\sqrt{10}$ if the loss fails to decrease over consecutive epochs with an additional early stopping criterion of 5 epochs with no improvement. The effective batch size is 32,768 tokens.

The number of parameters for all PSALM clan and family models are given in Table 3. All models were trained on four NVIDIA A100 80GB GPUs. To accommodate memory limitations on the GPU, all sequences are truncated to a maximum length of 4096 residues. This truncation strategy only applies to approximately 0.25% of sequences across the training and test sets and does not reflect a model limitation – PSALM can be used to annotate sequences of any length provided enough memory. All procedures from the HMMER tool suite use version 3.4 [4].

A.3 Model Capacity Experiments

We assess the model capacity of PSALM by evaluating performance across different model sizes, using the 8M, 35M, 150M, and 650M parameter ESM-2 models denoted as PSALM_8, PSALM_{35}, PSALM_{150}, and PSALM_{650} (Table 4). We find that PSALM's performance increases with ESM-2 model size up to the largest ESM-2 size tested (650M).

Table 4: PSALM model capacity results on MDPH-Bench

		Clan				Family			
PID	Model	TPR	FPR	F1	MCC	TPR	FPR	F1	MCC
0-20%	HMMER*	0.694	0.033	0.819	0.642	0.659	0.033	0.810	0.636
	PSALM_{650}	**0.944**	**0.022**	**0.985**	**0.957**	**0.750**	**0.012**	**0.978**	**0.947**
	PSALM_{150}	0.862	0.133	0.912	0.758	0.621	0.050	0.869	0.730
	PSALM_{35}	0.729	0.174	0.847	0.620	0.428	0.071	0.721	0.532
	PSALM_{8}	0.589	0.214	0.772	0.488	0.211	0.079	0.463	0.293
	PSALM_{OH}	0.490	0.100	0.764	0.559	0.089	0.022	0.236	0.203
20-40%	HMMER*	0.907	0.043	0.941	0.862	**0.876**	0.043	0.939	0.861
	PSALM_{650}	**0.966**	**0.020**	**0.985**	**0.964**	0.845	**0.015**	**0.982**	**0.959**
	PSALM_{150}	0.887	0.092	0.925	0.819	0.727	0.036	0.910	0.817
	PSALM_{35}	0.799	0.131	0.873	0.709	0.607	0.056	0.825	0.682
	PSALM_{8}	0.636	0.192	0.788	0.553	0.353	0.074	0.623	0.452
	PSALM_{OH}	0.516	0.107	0.780	0.602	0.102	0.023	0.282	0.251
40-60%	HMMER*	0.951	0.058	0.957	0.898	0.921	0.058	0.956	0.896
	PSALM_{650}	**0.977**	**0.020**	**0.986**	**0.966**	**0.924**	**0.017**	**0.984**	**0.964**
	PSALM_{150}	0.888	0.058	0.927	0.834	0.806	0.026	0.919	0.832
	PSALM_{35}	0.826	0.101	0.882	0.736	0.728	0.049	0.866	0.738
	PSALM_{8}	0.704	0.158	0.809	0.598	0.532	0.072	0.741	0.567
	PSALM_{OH}	0.666	0.104	0.835	0.671	0.159	0.029	0.430	0.363
60-80%	HMMER*	0.974	0.059	0.971	0.924	0.946	0.059	0.970	0.923
	PSALM_{650}	**0.984**	**0.018**	**0.988**	**0.970**	**0.957**	**0.016**	**0.988**	**0.968**
	PSALM_{150}	0.912	0.058	0.940	0.850	0.859	0.028	0.936	0.852
	PSALM_{35}	0.845	0.083	0.900	0.761	0.782	0.045	0.888	0.759
	PSALM_{8}	0.728	0.154	0.827	0.609	0.605	0.084	0.778	0.583
	PSALM_{OH}	0.788	0.094	0.890	0.745	0.216	0.027	0.573	0.478
80-100%	HMMER*	0.977	0.051	0.972	0.935	0.950	0.051	0.971	0.934
	PSALM_{650}	**0.981**	**0.015**	**0.986**	**0.969**	**0.967**	**0.012**	**0.986**	**0.968**
	PSALM_{150}	0.895	0.049	0.929	0.845	0.851	0.024	0.924	0.845
	PSALM_{35}	0.812	0.088	0.872	0.732	0.732	0.046	0.853	0.725
	PSALM_{8}	0.711	0.114	0.809	0.624	0.601	0.059	0.761	0.600
	PSALM_{OH}	0.877	0.066	0.925	0.836	0.282	0.018	0.709	0.630

A.4 Family-only PSALM Full Results

We conduct an additional ablation experiment in order to study the effect of predicting clan-level annotations as an interpretable intermediate in PSALM (Table 5). We retrain the PSALM family models without providing any predicted clan annotations and denote this model as PSALM_F for all ESM-2 model sizes tested in Appendix A.3. Clan predictions are generated by identifying the clan corresponding to the predicted family. We find that intermediate clan predictions are necessary for PSALM to surpass HMMER*.

Table 5: Family-only PSALM MDPH-Bench Results

		Clan				Family			
PID	Model	TPR	FPR	F1	MCC	TPR	FPR	F1	MCC
0-20%	HMMER*	0.694	0.033	**0.819**	0.642	**0.659**	0.033	**0.810**	0.636
	PSALM_F_{650}	**0.701**	**0.015**	0.827	**0.664**	0.632	**0.015**	0.811	**0.653**
	PSALM_F_{150}	0.630	0.034	0.781	0.596	0.540	0.034	0.753	0.576
	PSALM_F_{35}	0.560	0.065	0.733	0.523	0.412	0.065	0.670	0.476
	PSALM_F_{8}	0.394	0.115	0.599	0.355	0.232	0.115	0.469	0.253
20-40%	HMMER*	**0.907**	0.043	**0.941**	**0.862**	**0.876**	0.043	**0.939**	**0.861**
	PSALM_F_{650}	0.781	**0.011**	0.878	0.764	0.747	**0.011**	0.873	0.760
	PSALM_F_{150}	0.705	0.032	0.833	0.691	0.651	0.032	0.822	0.682
	PSALM_F_{35}	0.662	0.058	0.800	0.632	0.581	0.058	0.778	0.614
	PSALM_F_{8}	0.479	0.102	0.674	0.462	0.356	0.102	0.605	0.406
40-60%	HMMER*	**0.951**	0.058	**0.957**	**0.898**	**0.921**	0.058	**0.956**	**0.896**
	PSALM_F_{650}	0.833	**0.012**	0.906	0.810	0.816	**0.012**	0.904	0.809
	PSALM_F_{150}	0.785	0.025	0.877	0.759	0.758	0.025	0.873	0.756
	PSALM_F_{35}	0.746	0.048	0.846	0.703	0.708	0.048	0.839	0.697
	PSALM_F_{8}	0.594	0.087	0.749	0.557	0.520	0.087	0.723	0.535
60-80%	HMMER*	**0.974**	0.059	**0.971**	**0.924**	**0.946**	0.059	**0.970**	**0.923**
	PSALM_F_{650}	0.888	**0.012**	0.938	0.857	0.876	**0.012**	0.937	0.856
	PSALM_F_{150}	0.842	0.025	0.910	0.801	0.823	0.025	0.908	0.799
	PSALM_F_{35}	0.792	0.048	0.876	0.733	0.770	0.048	0.872	0.730
	PSALM_F_{8}	0.632	0.093	0.776	0.569	0.586	0.093	0.762	0.558
80-100%	HMMER*	**0.977**	0.051	**0.972**	**0.935**	**0.950**	0.051	**0.971**	**0.934**
	PSALM_F_{650}	0.892	**0.010**	0.940	0.875	0.887	**0.010**	0.939	0.875
	PSALM_F_{150}	0.835	0.024	0.903	0.808	0.819	0.024	0.901	0.806
	PSALM_F_{35}	0.740	0.047	0.839	0.701	0.720	0.047	0.835	0.697
	PSALM_F_{8}	0.661	0.078	0.783	0.609	0.627	0.078	0.774	0.601