Better Protein Function Prediction by Modeling Survivorship Bias

Zhongmou Chao [*1] **Poompol Buathong** [*2] **Ekaterina Selivanovitch** [1] **Susan Daniel** [1] **Peter I. Frazier** [3]

[*]Equal contribution [1]*Robert F. Smith School of Chemical and Biomolecular Engineering, Cornell University* [2]*Center for Applied Mathematics, Cornell University* [3]*School of Operations Research and Information Engineering, Cornell University*. Correspondence to: Poompol Buathong pb482@cornell.edu.

1. Introduction

Understanding the relationship between protein sequence and function is central to environmental sustainability, human health, and materials science [1, 2, 3]. However, existing protein datasets are not unbiased samples of sequence space: they are shaped by evolutionary and experimental selection processes that systematically overrepresent functional sequences while filtering out non-functional ones. This creates a survivorship bias that fundamentally limits standard supervised learning approaches.

This bias has two sources. First, natural evolution preferentially retains functional variants through selection, causing protein databases to be dominated by sequences that enhance organism survival [4, 5]. Second, experimental selection protocols such as biopanning systematically enrich for sequences with desired properties (e.g., binding affinity) while discarding those that fail to meet functional thresholds [6, 7]. As a result, protein datasets contain predominantly positive examples and few verified negatives.

This structure makes protein function prediction a positive-unlabeled (PU) learning problem [8], where observed sequences are functional (positive) and all others are unlabeled — they may be functional but unobserved, or truly non-functional. Existing PU methods address this setting by introducing a class prior: the probability that an unlabeled sequence is truly positive. However, these methods typically assume a constant class prior across all sequences, ignoring the fact that a sequence's probability of being observed depends on its evolutionary accessibility. This assumption is biologically unrealistic and limits model performance: sequences that are one mutation away from highly prevalent variants are far more likely to have been observed (if functional) than sequences requiring multiple simultaneous mutations.

We propose Evo-PU, a PU learning framework that models survivorship bias through an evolution-informed, sequence-dependent class prior. By modeling protein evolution at the nucleotide level, Evo-PU captures how mutational accessibility and prevalence jointly determine which functional sequences are observed. This yields a biologically grounded likelihood that interprets missing data correctly: absence of a mutationally accessible sequence is treated as evidence against functionality, while absence of an evolutionarily distant sequence is attributed to limited evolutionary reach rather than lack of function.

We evaluate Evo-PU on influenza viral protein data, using global surveillance sequences for training and independent laboratory mutagenesis assays for testing. Evo-PU improves identification of previously unseen peptide sequences associated with key viral functions—immune evasion, receptor binding, and membrane fusion—and outperforms or is competitive with existing PU learning methods, one-class classifiers, and deep generative models across multiple prediction tasks.

2. Evo-PU Framework

We now formalize the data-generating process for protein sequence observation and derive the Evo-PU likelihood. Section 2.1 introduces the probabilistic model and Section 2.2 relates it to existing methods.

2.1 Data-Generating Process and Likelihood

We model the process by which functional sequences become observed through mutation, selection, and surveillance. Let $x \in \mathcal{X}$ denote an amino acid sequence and let $A(x) \in \{0,1\}$ indicate whether x exhibits a functional property of interest. We aim to estimate the parameters θ in a probabilistic classifier $p_a(x;\theta) = P(A(x) = 1)$. Our approach is agnostic to the classifier used and supports any continuously-differentiable neural architecture.

In biological systems, amino acid sequences are produced by translating nucleotide sequences through the genetic code. Let $\mathcal{Y}$ denote the space of nucleotide sequences and let $T : \mathcal{Y} \to \mathcal{X}$ denote the biological translation map. For a given amino acid sequence x, we define $Y(x) = \{\, y \in \mathcal{Y} : T(y) = x \,\}$ to be the set of nucleotide sequences encoding x.

We use the term *emergence* to denote the event that a nucleotide sequence is generated by mutation. Let $E(y) \in \{0,1\}$ indicate whether nucleotide sequence y emerges. Let α be a vector of unknown nuisance parameters in a second probabilistic model $p_e(y;\alpha) = P(E(y) = 1)$. Details on the functional form of p_e are given in the appendix.

We let $O_Y(y) \in \{0,1\}$ indicate whether nucleotide sequence y is observed. We assume this occurs if y emerges, encodes a functional protein, and is detected by a surveillance process that has some unknown probability p_o of succeeding. Thus, $P(O_Y(y) = 1 \mid E(y), A(T(y))) = p_o\, E(y)\, A(T(y))$.

We let $O_X(x)$ indicate whether amino acid sequence x is observed. We have the relation $O_X(x) = \mathbb{I}\big(\exists\, y \in Y(x) \text{ such that } O_Y(y) = 1\big)$. Combining functionality, emergence, and observability yields

$$P(O_X(x) = 1) = p_a(x;\theta)\left[1 - \prod_{y \in Y(x)} \left(1 - p_o\, p_e(y;\alpha)\right)\right].$$

The second term on the right defines a sequence-dependent class prior corresponding to the probability that a functional sequence x is observed.

Given observed amino acid sequences $\mathcal{D}_n \subset \mathcal{X}$, we define the Evo-PU log-likelihood as

$$\sum_{x \in \mathcal{D}_n} \log P(O_X(x) = 1) + \sum_{x \notin \mathcal{D}_n} \log \big(1 - P(O_X(x) = 1)\big). \quad (1)$$

Exact evaluation of the Evo-PU likelihood is intractable because the size of sequence space is exponential in sequence length. We therefore approximate the likelihood by restricting attention to a biologically plausible subset of unobserved sequences, $x \notin \mathcal{D}_n$, that are generated via nucleotide-level mutation from observed data. This sampling-based approximation focuses computation of the second summation term in Eq (1) on sequences with non-negligible emergence probability and enables efficient training.

We train the classifier to estimate θ and the nuisance parameters α, p_o by maximizing this approximated log-likelihood (1).

2.2 Comparison to Previously-Proposed Likelihoods

A key feature of Evo-PU is that it interprets absence of observation differently depending on mutational accessibility: for sequences close to observed variants, non-observation provides evidence against functionality, whereas for evolutionarily distant sequences it reflects limited evolutionary reach. This contrasts with existing methods that make uniform assumptions about observability.

Evo-PU generalizes two special cases. First, if all functional sequences are always observed ($p_o p_e(y; \alpha) = 1$), Eq. (1) reduces to standard binary classification, which treats all unobserved sequences as negative regardless of their mutational accessibility. Second, if observation probability is constant across sequences ($\left[1 - \prod_{y \in Y(x)} (1 - p_o\, p_e(y; \alpha))\right] := q$ for some q), the likelihood recovers the Protein-PU formulation of [9], which uses a sequence-independent class prior. Evo-PU relaxes both assumptions by modeling observability as sequence-dependent and evolution-informed.

3. Numerical Experiments and Results

We evaluate Evo-PU on influenza classification tasks (fusion, binding, and evasion), using global surveillance data [10, 11] for training and independent experimental mutagenesis studies for testing [12, 13, 14, 15, 16, 17, 18, 19, 20, 21].

We compare Evo-PU with one-class classifiers (OC-SVM [22], iForest [23]), PU learning methods (2Step [8], Protein-PU [9]), and protein language model baselines (k-NN × ESM2 [24], zero-shot [25], EVE [26]).

For PU methods, unlabeled data are generated via simple random sampling. For k-NN × ESM2, the same generated sequences are included in training and treated as negative examples. We further conducted an ablation study on the unlabeled data generation strategy, where unlabeled sequences are generated using the same procedure as that used to approximate the log-likelihood, which we call E_GEN, which produces biologically plausible sequences with non-negligible emergence probability, as described in Section 2.1. The results are presented in the Appendix. Performance is evaluated using AUC metric.

Evo-PU can be used with any probabilistic classifier. We consider logistic regression (LR) as a simple baseline and a neural network inspired by the Wide & Deep architecture (WD) [27] as a more expressive model.

Tables 1 and 2 present the results when the WD network and the LR model are used as a classifier, respectively. Across both classifiers, Evo-PU consistently outperforms all competing methods on Fusion and Binding tasks and is competitive with Protein-PU and OCC methods on Evasion task. These results demonstrate the benefits of incorporating evolutionary information into the training process, highlighting its potential to accelerate protein design and related applications.

Table 1: AUC comparison across three tasks: Fusion, Binding, and Evasion, using a wide deep (WD) network as the probabilistic classifier. Bold text denotes the best performances. Evo-PU performs best on two tasks and has an AUC within .03 of the best on the third task.

Methods	AUC (WD)		
	Fusion	Binding	Evasion
Evo-PU	**0.66**	**0.89**	0.96
Protein-PU	0.64	0.69	**0.99**
2Step	0.51	0.5	0.78
iForest	0.59	0.77	**0.98**
OC-SVM	0.62	0.67	**0.98**
k-NN × ESM2	0.50	0.51	0.76
ESM-1v (zero shot)	0.63	0.33	0.32
EVE	0.61	0.33	0.59

Table 2: AUC comparison across the same three tasks as Table 1, using a logistic regression (LR) as a simple baseline probabilistic classifier. Bold text denotes the best performances. Evo-PU performs best on two tasks and has an AUC within .02 of the best on the third task.

Methods	AUC (LR)		
	Fusion	Binding	Evasion
Evo-PU	**0.69**	**0.83**	0.96
Protein-PU	0.62	0.70	**0.98**
2Step	0.52	0.55	0.92
iForest	0.59	0.77	**0.98**
OC-SVM	0.62	0.67	**0.98**
k-NN × ESM2	0.50	0.51	0.76
ESM-1v (zero shot)	0.63	0.33	0.32
EVE	0.61	0.33	0.59

Appendix

Nucleotide emergence model.

The emergence probability $p_e(y;\alpha)$ captures how likely an unobserved nucleotide sequence y' is to arise and persist. The key insight is that y' can emerge through multiple mutational pathways from any prevalent sequence in the observed data, and more prevalent sequences contribute more opportunities for mutation. For an unobserved nucleotide sequence y', we define

$$p_e(y';\alpha) = 1 - \prod_{y_p \in \mathcal{D}_N^p} \left(1 - P(y_p \to y')\,\alpha\right)^{c(y_p)}, \qquad (2)$$

where $\mathcal{D}_N^p$ is the set of previously observed sequences, $P(y_p \to y')$ is the mutation probability from y_p to y', $c(y_p)$ is the prevalence of y_p, and α is the probability that a mutated sequence successfully establishes. This formulation reflects that higher prevalence and closer mutational distance both increase emergence probability. For sequences already observed, we set $p_e(y;\alpha) = 1$.

Impact of unlabeled data generation

Evo-PU uses a method, described in Section 2.1, for approximating the log-likelihood (1) that restricts attention to a biologically-plausible subset of unobserved sequences. These are obtained via nucleotide-level mutation from observed sequences. This method, which we call E_Gen, is of independent interest, both as a possible source of Evo-PU's performance, and because it has potential utility for a component of standard PU learning approaches: unlabeled data generation.

Indeed, standard PU learning approaches applied to protein function prediction must generate unobserved sequences to constitute the pool of unlabeled instances because the full space of unobserved sequences is too large. A standard approach is to generate these uniformly at random. Here, we consider instead generating them using E_Gen.

We explore E_Gen in the context of three PU methods that rely on unlabeled data generation: Protein-PU, 2Steps and k-NN × ESM2. For each such method, across the three different prediction tasks (Fusion, Binding, Evasion) and the two probabilistic classifiers (logistic regression and a wide deep network model) studied above, we compare data generation using uniform random sampling (Random) against E_Gen.

Tables 3 and 4 present the results. For each task and PU method, we boldface the data generation method that performs best. We found that overall using E_Gen method improves the performance of these approaches on Fusion task, yielding higher AUC values, but worsens the performances of the approaches on Binding and Evasion tasks. Overall, we see E_Gen as being helpful in some situations but not being the primary source of Evo-PU's overall performance boost relative to other methods.

Note that the results for k-NN × ESM2 are the same between LR and WD. This is because k-NN × ESM2 uses a k-NN classifier rather than LR or WD and so its performance is insensitive to this choice.

We also find that Protein-PU on the Fusion task using E_Gen performs particularly well, performing the best out of all methods considered (even better than Evo-PU). It is worth noting Protein-PU using E_Gen is similar in implementation to Evo-PU — using a uniform class prior on unlabeled data that were selected because they had a large value for Evo-PU's class prior is similar to using Evo-PU's class prior directly.

References

[1] Vincent Tournier, Sophie Duquesne, Frédérique Guillamot, Henri Cramail, Daniel Taton, Alain Marty, and Isabelle André. Enzymes' power for plastics degradation. *Chemical Reviews*, 123(9):5612–5701, 2023.

[2] Kathryn E Tiller and Peter M Tessier. Advances in antibody design. *Annual review of biomedical engineering*, 17(1):191–216, 2015.

[3] Wei Wei, Luigi Petrone, YerPeng Tan, Hao Cai, Jacob N Israelachvili, Ali Miserez, and J Herbert Waite. An underwater surface-drying peptide inspired by a mussel adhesive protein. *Advanced functional materials*, 26(20):3496–3507, 2016.

[4] Luis Bermúdez-Guzmán, Gabriel Jimenez-Huezo, Andrés Arguedas, and Alejandro Leal. Mutational survivorship bias: The case of pnkp. *PLoS One*, 15(12):e0237682, 2020.

[5] Adam Thomas, Benjamin D Evans, Mark van der Giezen, and Nicholas J Harmer. Survivor bias drives overestimation of stability in reconstructed ancestral proteins. *bioRxiv*, pages 2022–11, 2022.

[6] Ricardo J Giordano, Marina Cardó-Vila, Johanna Lahdenranta, Renata Pasqualini, and Wadih Arap. Biopanning and rapid analysis of selective interactive ligands, 2001.

[7] Michael J McGuire, Shunzi Li, and Kathlynn C Brown. Biopanning of phage displayed peptide libraries for the isolation of cell-specific ligands. *Methods Mol Biol*, 504:291–321, 2009.

[8] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760, 2020.

[9] Hyebin Song, Bennett J Bremer, Emily C Hinds, Garvesh Raskutti, and Philip A Romero. Inferring protein sequence-function relationships with large-scale positive-unlabeled learning. *Cell systems*, 12(1):92–101, 2021.

[10] NCBI. Influenza virus data hub, 2024. `https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/virus?SeqType_s=Nucleotide&VirusLineage_ss=taxid:197911&VirusLineage_ss=taxid:`

Table 3: AUC comparison for a study of E_Gen (an unlabeled data generation method) across three tasks—Fusion, Binding, and Evasion—using logistic regression (LR) as the probabilistic classifier.

Methods	Unlabeled Data Generation Methods	AUC (LR)		
		Fusion	Binding	Evasion
Protein-PU	Random	0.62	**0.70**	**0.98**
	E_Gen	**0.67**	0.55	0.80
2Step	Random	0.52	**0.55**	**0.92**
	E_Gen	**0.62**	0.34	0.47
k-NN × ESM2	Random	0.50	0.51	0.76
	E_Gen	**0.62**	**0.55**	**0.83**

Table 4: AUC comparison for a study of E_Gen (an unlabeled data generation method) across the same three tasks as Table 3 using a wide and deep (WD) network as the probabilistic classifier.

Methods	Unlabeled Data Generation Methods	AUC (WD)		
		Fusion	Binding	Evasion
Protein-PU	Random	0.64	**0.69**	**0.99**
	E_Gen	**0.71**	0.66	0.78
2Step	Random	**0.51**	**0.50**	**0.78**
	E_Gen	0.50	**0.50**	0.68
k-NN × ESM2	Random	0.50	0.51	0.76
	E_Gen	**0.62**	**0.55**	**0.83**

197912&VirusLineage_ss=taxid:197913&VirusLineage_ss=taxid:1511083 [Accessed: (2024-10-11)].

[11] Yuelong Shu and John McCauley. Gisaid: Global initiative on sharing all influenza data–from vision to reality. *Eurosurveillance*, 22(13):30494, 2017.

[12] Xing Han, David A Steinhauer, Stephen A Wharton, and Lukas K Tamm. Interaction of mutant influenza virus hemagglutinin fusion peptides with lipid bilayers: probing the role of hydrophobic residue size in the central region of the fusion peptide. *Biochemistry*, 38(45):15052–15059, 1999.

[13] Hui Qiao, R Todd Armstrong, Grigory B Melikyan, Fredric S Cohen, and Judith M White. A specific point mutant at position 1 of the influenza hemagglutinin fusion peptide displays a hemifusion phenotype. *Molecular biology of the cell*, 10(8):2759–2769, 1999.

[14] Lukas K Tamm, Xing Han, Yinling Li, and Alex L Lai. Structure and function of membrane fusion peptides. *Peptide Science: Original Research on Biomolecules*, 66(4):249–260, 2002.

[15] Alex L Lai, Heather Park, Judith M White, and Lukas K Tamm. Fusion peptide of influenza hemagglutinin requires a fixed angle boomerang structure for activity. *Journal of Biological Chemistry*, 281(9):5760–5770, 2006.

[16] Y Su, Xingguo Zhu, Y Wang, M Wu, and P Tien. Evaluation of glu11 and gly8 of the h5n1 influenza hemagglutinin fusion peptide in membrane fusion using pseudotype virus and reverse genetics. *Archives of virology*, 153:247–257, 2008.

[17] Karen J Cross, William A Langley, Rupert J Russell, John J Skehel, and David A Steinhauer. Composition and functions of the influenza fusion peptide. *Protein and peptide letters*, 16(7):766–778, 2009.

[18] Zhi-Yong Yang, Chih-Jen Wei, Wing-Pui Kong, Lan Wu, Ling Xu, David F Smith, and Gary J Nabel. Immunization by avian h5 influenza hemagglutinin mutants with altered receptor binding specificity. *Science*, 317(5839):825–828, 2007.

[19] Javier Martín, Stephen A Wharton, Yi Pu Lin, Darin K Takemoto, John J Skehel, Don C Wiley, and David A Steinhauer. Studies of the binding properties of influenza hemagglutinin receptor-site mutants. *Virology*, 241(1):101–111, 1998.

[20] Taronna R Maines, Li-Mei Chen, Neal Van Hoeven, Terrence M Tumpey, Ola Blixt, Jessica A Belser, Kortney M Gustin, Melissa B Pearce, Claudia Pappas, James Stevens, et al. Effect of receptor binding domain mutations on receptor binding and transmissibility of avian influenza h5n1 viruses. *Virology*, 413(1):139–147, 2011.

[21] Li-Mei Chen, Ola Blixt, James Stevens, Aleksandr S Lipatov, Charles T Davis, Brian E Collins, Nancy J Cox, James C Paulson, and Ruben O Donis. In vitro evolution of h5n1 avian influenza virus toward human-type receptor specificity. *Virology*, 422(1):105–113, 2012.

[22] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional

distribution. *Neural computation*, 13(7):1443–1471, 2001.

[23] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[24] Farzaneh Esmaili, Yongfang Qin, Duolin Wang, and Dong Xu. Kinase-substrate prediction using an autoregressive model. *Computational and Structural Biotechnology Journal*, 27:1103–1111, 2025.

[25] Joshua Meier, Roshan Rao, Robert Verkuil, Jason Liu, Tom Sercu, and Alex Rives. Language models enable zero-shot prediction of the effects of mutations on protein function. *Advances in neural information processing systems*, 34:29287–29303, 2021.

[26] Jonathan Frazer, Pascal Notin, Mafalda Dias, Aidan Gomez, Joseph K Min, Kelly Brock, Yarin Gal, and Debora S Marks. Disease variant prediction with deep generative models of evolutionary data. *Nature*, 599(7883):91–95, 2021.

[27] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10, 2016.