PRECISION INVERSE FOLDING: FROM LOW-ORDER MUTANT MEASUREMENTS TO HIGHER-ORDER VARIANT PRIORITIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein inverse folding models optimize sequence–structure compatibility but provide limited guidance on protein function. In practical protein engineering, only a small number of low-order mutants can typically be experimentally characterized, while promising designs may lie in a much larger space of higher-order combinations. We study this *two-to-many* setting, using only wild-type, single, and double-mutant wet-lab measurements to prioritize unseen variants with three or more mutations. Our approach constructs cross-order preferences from measured low-order variants and synthetic higher-order negatives derived from inactivity-associated mutations, focuses alignment on function-relevant mutations through critical-site preference optimization, and enriches structural representations with protein foundation-model embeddings. Across four protein fitness datasets and two inverse-folding architectures, our approach consistently improves higher-order variant prioritization, demonstrating that limited low-order experimental data can effectively guide exploration beyond the experimentally accessible mutational regime.

1 INTRODUCTION

Protein engineering campaigns aim to discover variants with improved function—such as higher catalytic activity, tighter binding, or greater stability—starting from a wild-type sequence and a target structure. In practice, this search is bottlenecked by wet-lab characterization: each candidate must be expressed and assayed before its function is known, while the number of possible multi-site variants grows combinatorially with the number of mutated positions (Wu et al., 2019; Yang et al., 2019). Consequently, early-stage campaigns typically characterize only a small, low-order neighborhood around the wild type (Wu et al., 2016), leaving the much larger space of higher-order combinations to be prioritized computationally.

Structure-conditioned inverse folding (IF) models such as ProteinMPNN (Dauparas et al., 2022) and ESM-IF (Hsu et al., 2022) provide powerful sequence-design priors by learning which sequences are compatible with a given backbone. However, their training objective is primarily structural rather than functional: sequences with comparable structural compatibility can exhibit substantially different functional activity (Jayaraman et al., 2022). Recent preference-alignment approaches have begun to bridge this gap by steering protein models toward variants with desirable functional properties (Tamatgar et al., 2026; Widatalla et al., 2026). ProteinDPO (Widatalla et al., 2026) leverages a large collection of experimentally measured fitness data to construct preference supervision, whereas function-aware preference alignment (FPA) (Tamatgar et al., 2026) constructs preference pairs entirely *in silico*. However, neither explicitly studies whether experimental measurements from low-order mutants can be leveraged to guide the design and prioritization of higher-order variants beyond the observed mutational regime.

This motivates a practically important *two-to-many* setting: can measurements from wild-type, single-, and double-mutant variants be used to prioritize functional variants carrying three or more mutations? Double mutants are particularly informative because they expose pairwise epistatic interactions absent from single-mutant measurements, and recent evidence suggests that such low-order supervision can support extrapolation to substantially higher-order combinations (Tran et al., 2026). Yet this setting

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

introduces three challenges. First, supervision and evaluation lie in different mutational regimes. Second, limited low-order measurements provide only sparse informative preferences. Third, standard IF representations are optimized primarily for sequence–structure compatibility rather than functional discrimination. Addressing these challenges therefore requires not only preference supervision that extrapolates across mutation orders, but also an objective and representation space suited to extracting functional information from sparse measurements.

In this work, we study *two-to-many protein inverse folding* using the FLIP2 *number split* (Didi et al., 2026): models learn only from wild-type, single-, and double-mutant measurements and are evaluated on held-out variants carrying three or more mutations. We construct cross-order preferences by combining experimentally grounded low-order comparisons with higher-order negatives recombined from inactivity-associated mutations, extending supervision beyond the assayed regime without additional wet-lab labels. To improve functional discrimination and cross-order generalization, we further augment structure-conditioned IF representations with residue-level embeddings from a pretrained protein foundation model, providing complementary sequence-context information beyond backbone compatibility. Preference optimization is performed over function-relevant mutational sites. Finally, we cast evaluation as *higher-order retrieval*, directly assessing whether the model can prioritize functional variants beyond the observed mutational regime.

The contributions of our work are summarized as follows:

- We formulate *two-to-many protein inverse folding*, where models learn from wild-type, single-, and double-mutant measurements but prioritize high-activity variants with three or more mutations.
- We develop a cross-order preference construction strategy that combines measured low-order comparisons with higher-order negatives recombined from inactivity-associated mutations, extending supervision beyond the assayed regime without additional wet-lab labels.
- We augment structure-conditioned inverse-folding representations with residue-level embeddings from a pretrained protein foundation model, providing complementary sequence-context information for functional discrimination and cross-order generalization.
- Across four FLIP2 fitness datasets and two inverse-folding architectures, our approach consistently improves higher-order variant prioritization over pretrained baselines, demonstrating meaningful transfer of low-order functional supervision across mutation counts.

2 Related Work

Structure-Conditioned Inverse Folding ProteinMPNN (Dauparas et al., 2022) and ESM-IF (Hsu et al., 2022), the two base architectures used in this work represent two complementary design choices for structure-conditioned sequence generation: a message-passing network over a k-nearest-neighbor backbone graph in the former, and a GVP-Transformer trained in part on AlphaFold2-predicted structures at scale in the latter. Subsequent work has extended this family along several axes, including conditioning on small-molecule ligands and cofactors (Dauparas et al., 2025), joint structure-sequence generation (Watson et al., 2023), and programmable generative modeling of both sequence and structure (Ingraham et al., 2023).

Preference Alignment for Protein Design Direct Preference Optimization (DPO) (Rafailov et al., 2023) reframes reinforcement-learning-style alignment as a supervised classification loss over preferred/dispreferred pairs, avoiding an explicit reward model. This has recently been adapted to protein design: ProteinDPO (Widatalla et al., 2026) aligns a structure-conditioned generative model to experimentally measured stability using pairs constructed from a large mutational stability dataset. (Tamatgar et al., 2026) took a complementary approach, constructing preference pairs *in silico* via hypothesis-driven perturbation of functionally critical residues, and fine-tuning IF models to favor function-preserving over function-disrupting sequences, without requiring additional wet-lab measurements. The present work builds on this function-aware alignment framework by augmenting the FiLM-based (Perez et al., 2018) functional conditioning with a preference-pair construction strategy suited to the low-order-to-high-order extrapolation setting (Section 3.2).

Beyond direct preference optimization, other lines of work steer protein generative models toward functional objectives using reinforcement learning against scalar or structural reward signals, such

as predicted structural confidence, thermostability, or activity oracles. These approaches typically require an external reward model or oracle available at training time; our approach instead relies only on preference pairs constructed from existing labeled low-order mutants without a separately trained reward model.

Extrapolation Beyond Low-Order Mutants Whether models or experimental strategies trained on low-order mutants can inform higher-order combinations is closely tied to the structure of epistasis in the underlying fitness landscape. Early combinatorially complete landscape studies found that while pairwise epistasis can block direct adaptive paths, such traps are often circumvented via indirect paths through genotype space, indicating that higher-fitness combinations remain reachable even when local, low-order effects are individually unfavorable (Wu et al., 2016). More directly relevant to our setting, MULTI-evolve (Tran et al., 2026) recently demonstrated, across several distinct proteins with wet-lab validation, that models trained primarily on single- and double-mutant data can guide the discovery of substantially higher-order multi-mutants with large functional gains, indicating that compact, low-order sampling can support meaningful extrapolation when the fitness landscape is reasonably well approximated by low-order interactions. We build on the premise that low-order supervision is informative beyond its own mutational neighborhood and study it through the lens of preference-aligned inverse folding rather than iterative wet-lab evolution.

ProSpero (Kmicikiewicz et al., 2025) addresses a related but distinct generalization challenge: designing sequences that explore beyond the wild-type neighborhood via active learning, using a surrogate fitness model updated from oracle feedback to guide a frozen pretrained generative model through biologically-constrained sampling. ProSpero and our work share the broader goal of extrapolating beyond the low-order mutants that are cheapest to assay, but differ in mechanism: ProSpero performs iterative, oracle-guided exploration at generation time, whereas we study whether preference alignment trained once, on a fixed pool of low-order mutants, transfers its learned ranking to higher-order mutants at evaluation time without further oracle queries.

3 Precision Inverse Folding from Low-Order Experimental Supervision

In this section, we propose *Precision Inverse Folding (PIF)*, a preference-alignment framework for extrapolating functional supervision from experimentally accessible low-order mutants to unseen higher-order combinatorial variants.

3.1 Two-to-Many Protein Inverse Folding

Protein IF aims to recover or design an amino-acid sequence that is compatible with a given protein backbone. Let S denote a target backbone and $x = (x_1, \ldots, x_L)$ a protein sequence of length L. A structure-conditioned inverse-folding model defines the sequence probability autoregressively as

$$\pi_\theta(x \mid S) = \prod_{i=1}^{L} p_\theta(x_i \mid x_{<i}, S), \tag{1}$$

where $x_{<i} = (x_1, \ldots, x_{i-1})$ denotes the previously generated residues.

Existing IF models are typically optimized for sequence recovery under structural conditioning. This provides a strong prior for sequence–structure compatibility, but does not explicitly distinguish sequences by biological function. As a result, variants that are similarly compatible with the same backbone can exhibit substantially different activities (Tamatgar et al., 2026; Widatalla et al., 2026). Incorporating experimentally measured fitness provides a natural source of functional supervision, but such measurements are expensive and therefore cover only a small fraction of the combinatorial sequence space. In practice, experimental characterization is often concentrated around a known wild-type sequence, whereas higher-order mutation combinations remain largely unmeasured.

Particularly, let $d(\cdot, \cdot)$ be the Hamming distance between two sequences and y_i be the experimentally measured property of variant x_i. We define $\mathcal{D}_{\text{low}} = \{(x_i, y_i) : d(x_i, x^{\text{wt}}) \leq 2\}$ containing only wild-type, single-mutant, and double-mutant measurements, and $\mathcal{D}_{\text{high}} = \{(x_i, y_i) : d(x_i, x^{\text{wt}}) \geq 3\}$ whose label y_i is withheld and used exclusively for evaluation. We instantiate this setting using the

FLIP2 *number split* (Didi et al., 2026) (Appendix A), where training and validation contain only low-order variants, while variants with three or more mutations are reserved for testing. This induces an explicit mutation-order shift between the experimentally characterized region used for alignment and the higher-order sequence space in which the model must generalize.

We refer to this setting as *two-to-many protein inverse folding*. Starting from a pretrained inverse-folding model $\pi_{\theta_0}(x \mid S)$, we use only the functional supervision in $\mathcal{D}_{\text{low}}$ to obtain an aligned model $\pi_\theta(x \mid S)$ whose likelihood better reflects experimentally relevant functional preferences while retaining structural compatibility. Specifically, for two variants (x_i, y_i) and (x_j, y_j) conditioned on the same backbone S, we seek

$$y_i > y_j \quad \Longrightarrow \quad \log \pi_\theta(x_i \mid S) > \log \pi_\theta(x_j \mid S). \tag{2}$$

The objective is therefore not to learn a separate fitness predictor or to regress the complete fitness landscape. Instead, we directly align the inverse-folding distribution with functional preferences, so that sequence–structure compatibility and functional activity become better coupled within the generative model.

Accordingly, evaluation focuses on the ranking induced by $\pi_\theta(x \mid S)$ over $\mathcal{D}_{\text{high}}$, with particular emphasis on whether variants with desirable measured properties are concentrated near the top of the ranking. This reflects the practical protein-engineering setting in which only a limited number of candidates can be selected for experimental validation.

Achieving this objective requires addressing three coupled challenges: *(i)* functional preferences learned from low-order mutants must extrapolate across mutation orders; *(ii)* low-order measurements provide only sparse preference supervision for the much larger higher-order space; and *(iii)* conventional inverse-folding representations are optimized primarily for sequence–structure compatibility rather than fine-grained discrimination of functional and biophysical differences.

We address these challenges through cross-order preference construction and protein-foundation-model-conditioned representation adaptation.

3.2 Cross-Order Preference Construction

To directly incorporate experimental fitness measurements into IF alignment, we first convert the measured low-order variants into pairwise preference data. However, such preferences remain restricted to the low order-mutant regime observed during training. We therefore construct two complementary sources of preference supervision: experimentally grounded low-order preferences and synthetic higher-order negative preferences.

Experimentally grounded preferences. For two measured variants $(x_i, y_i), (x_j, y_j) \in \mathcal{D}_{\text{low}}$, we construct a preference pair whenever their measured activities are sufficiently separated:

$$\mathcal{P}(D_{\text{low}}) = \{(x_i, x_j) \mid (x_i, y_i), (x_j, y_j) \in \mathcal{D}_{\text{low}}, y_i - y_j > \delta\}, \tag{3}$$

where δ removes near-ties that may be sensitive to experimental noise. This transformation allows the experimentally measured functional differences to be used directly for preference optimization of the IF model, while retaining the pretrained model as the structure-conditioned reference distribution.

Higher-order negative preferences. Preferences in $\mathcal{P}(D_{\text{low}})$ alone provide no direct supervision beyond the measured mutational neighborhood. To expose the model to the higher-order regime without requiring additional assays, we construct synthetic negative sequences from mutations associated with low activity in the training data.

Specifically, we first identify an inactive subset from the training split and extract substitutions that are enriched in these low-activity variants. Mutations that also occur broadly among active training variants are filtered out, and the remaining inactivity-associated substitutions form a mutation pool $\mathcal{M}^-$. We also define $\mathcal{D}^+_{\text{low}} \subseteq \mathcal{D}_{\text{low}}$ as the subset of experimentally measured low-order variants with high activity. The dataset-specific criteria for constructing the inactive and high-activity subsets, together with the mutation-pool construction, are provided in Appendix A.7 and Appendix A.8, respectively.

Using $\mathcal{M}^-$, we construct higher-order negatives through two complementary recombination strategies. First, multiple inactivity-associated mutations are introduced directly onto the wild-type background:

$$X^-_{\mathrm{wt}} = \mathrm{Mutate}\left(x^{\mathrm{wt}}; \{m_i\}_{i=1}^r\right), \qquad \{m_i\}_{i=1}^r \subseteq \mathcal{M}^-, \quad r \geq 3. \tag{4}$$

This produces synthetic higher-order variants composed of inactivity-associated mutations.

Second, inactivity-associated mutations are introduced into experimentally measured high-activity low-order variants $(x^+, y^+) \in \mathcal{D}^+_{\mathrm{low}}$,

$$X^-_{x^+} = \mathrm{Mutate}\left(x^+; \{m_i\}_{i=1}^r\right), \qquad \{m_i\}_{i=1}^r \subseteq \mathcal{M}^-,\ r \geq 3 - d(x^+, x^{\mathrm{wt}}). \tag{5}$$

yielding harder negatives located closer to experimentally favorable regions of sequence space.

Because the functional effects of recombinant mutations are not quantitatively known and may depend on epistatic interactions, we do not assign pseudo-fitness values to these synthetic sequences. Instead, they are used only to define ordinal constraints against experimentally observed high-activity variants:

$$\mathcal{P}_{\mathrm{neg}} = \left\{(x^+, \tilde{x}^-) \mid (x^+, y^+) \in \mathcal{D}^+_{\mathrm{low}}, \tilde{x}^- \in X^-_{\mathrm{wt}} \cup X^-_{x^+}\right\}, \tag{6}$$

where X^-_{wt} and $X^-_{x^+}$ are the synthetic higher-order negative sets generated by the two recombination strategies described above.

The resulting preference set $\mathcal{P}_{\mathrm{ref}} = \mathcal{P}(D_{\mathrm{low}}) \cup \mathcal{P}_{\mathrm{neg}}$ therefore combines experimentally grounded low-order preferences with synthetic higher-order ordinal constraints. The former anchors alignment to measured functional differences, while the latter expands supervision into the unseen mutational regime without requiring additional wet-lab measurements.

3.3 Protein-Foundation-Model-Enriched Representation Adaptation

Even with cross-order preference supervision, effective functional alignment remains constrained by the representations inherited from pretrained inverse-folding models. ProteinMPNN (Dauparas et al., 2022) and ESM-IF (Hsu et al., 2022) learn strong structure-conditioned representations, but these representations are optimized primarily for sequence recovery rather than functional discrimination. We therefore enrich their structural representations with residue-level sequence embeddings from the pretrained protein foundation model (Lin et al., 2023).

Let h_i denote the structural representation at residue i and e_i its residue-level sequence embedding. A lightweight FiLM-like module (Perez et al., 2018) projects e_i into modulation parameters (γ_i, β_i) and adapts the structural representation as

$$h'_i = h_i \odot [1 + \sigma(g)\tanh(\gamma_i)] + \sigma(g)\beta_i,$$

where g is a learnable gate controlling the strength of foundation-model conditioning.

Rather than replacing the pretrained structural representation, this design preserves the original structural pathway while adaptively injecting sequence-context information learned from large-scale protein sequences. The same module is applied at the encoder–decoder boundary of ProteinMPNN and ESM-IF, providing richer representations for functional preference alignment.

3.4 Critical-site Preference Optimization

Low-order variants typically differ at only a few residues, so computing preference likelihoods over the entire sequence can dilute the functional signal with contributions from many unchanged positions. We therefore restrict preference optimization to the critical-site set $F(x^+, x^-) = \{i : x^+_i \neq x^-_i\}$.

For a sequence x, we define its average log-likelihood over the critical sites as $\bar{\ell}^F_\theta(x \mid S) = \frac{1}{|F|}\sum_{i\in F} \log p_\theta(x_i \mid x_{<i}, S)$. Based on the pairwise preference optimization objective in Wu et al. (2025b), we define Critical-site Preference Optimization (CPO) using these localized likelihoods following (Tamatgar et al., 2026). Given a frozen reference inverse-folding model π_{ref},

$$\mathcal{L}_{\mathrm{CPO}} = \mathbb{E}_{(x^+,x^-)\sim\mathcal{D}_{\mathrm{pref}}}[(a - \tfrac{1}{2})^2 + (b + \tfrac{1}{2})^2], \tag{7}$$

where $a = \beta[\bar{\ell}^F_\theta(x^+ \mid S) - \bar{\ell}^F_{\mathrm{ref}}(x^+ \mid S)]$ and $b = \beta[\bar{\ell}^F_\theta(x^- \mid S) - \bar{\ell}^F_{\mathrm{ref}}(x^- \mid S)]$.

This formulation concentrates preference alignment on the residues that distinguish the preferred and dispreferred variants, rather than allowing the many unchanged positions to dominate the sequence-level likelihood. This is particularly important in our setting, where experimentally measured supervision is primarily derived from single- and double-mutant variants.

To preserve the general sequence-design capability of the pretrained inverse-folding model, we jointly optimize CPO with its original sequence-recovery objective $\mathcal{L}_{\mathrm{CE}}$, computed on the corresponding structure–sequence training corpus. During this forward pass, the foundation-model conditioning module is disabled, so that the reconstruction objective regularizes the original structure-conditioned inverse-folding pathway without access to the target sequence embedding.

For a categorical dataset, whose activity is represented by K discrete categories rather than a continuous score, we additionally introduce a lightweight classification head g_{cls} on top of the decoder's joint structure-sequence representation. Given the foundation-model grounded representations $h'(x)$ (described in Section 3.3), we predict the activity-category logits as $g_{\mathrm{cls}}(h'(x)) \in \mathbb{R}^K$.

Because synthetic higher-order negatives do not have experimentally measured category labels, the classification objective is applied only to measured samples from the categorical dataset. Let $\mathcal{D}_{\mathrm{low}}^{\mathrm{cat}}$ denote the labeled low-order training set of the categorical dataset. We define

$$\mathcal{L}_{\mathrm{cls}} = \mathbb{E}_{(x,c)\sim\mathcal{D}_{\mathrm{low}}^{\mathrm{cat}}} \left[\mathrm{CE}\left(g_{\mathrm{cls}}(h'(x)), c\right) \right].$$

The complete training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{CPO}} + \lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \mathbb{I}_{\mathrm{cat}}\lambda_{\mathrm{cls}}\mathcal{L}_{\mathrm{cls}}, \qquad \text{where} \quad \mathbb{I}_{\mathrm{cat}} = \begin{cases} 1, & \text{for cat,} \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

At inference time, unseen higher-order variants are ranked using their length-normalized conditional log-likelihood under the aligned inverse-folding model: $s_\theta(x \mid S) = \frac{1}{L}\log \pi_\theta(x \mid S) = \frac{1}{L}\sum_{i=1}^{L} \log p_\theta(x_i \mid x_{<i}, S)$. Critical-site likelihoods are used only during preference optimization because $F(x^+, x^-)$ is defined relative to a preference pair. At inference time, each candidate is evaluated independently, so the normalized full-sequence likelihood provides a consistent scoring function for higher-order variant prioritization.

The complete training procedure of Precision Inverse Folding (PIF) is summarized in Algorithm 1.

4 Experiment

We evaluate whether limited functional measurements from low-order mutants can effectively guide the prioritization of unseen higher-order variants. Our experiments address two questions: *(i)* whether low-order supervision improves higher-order retrieval and *(ii)* how the proposed synthetic higher-order negatives and foundation-model-enriched representations contribute to this improvement.

4.1 Experimental Setup

We fine-tune two widely used inverse-folding backbones ProteinMPNN (Dauparas et al., 2022) and ESM-IF (Hsu et al., 2022) on four FLIP2 datasets (see Appendix A). For both architectures, residue-level conditioning features are extracted from the final layer of ESM2-650M (Lin et al., 2023). Full architecture, optimization, and hyperparameter details are provided in Appendix B.1 and B.4.

Baselines. We evaluate PIF on the vanilla ProteinMPNN (`v_48_020`) and ESM-IF (`esm_if1_gvp4_t16_142M_UR50`) backbones. For each backbone, we compare against SPPO (Wu et al., 2025b), as a general preference-based fine-tuning method, and FPA (Tamatgar et al., 2026), a state-of-the-art function-aware preference alignment method for protein inverse folding. To isolate the contributions of PIF, we further evaluate two ablations: PIF w/o Neg, which removes the pool-based cross-order negatives and uses only preference pairs constructed from measured low-order variants, and PIF w/o PFM, which removes the ESM2-conditioned FiLM module. Candidate variants are ranked by the normalized conditional log-likelihood assigned by each corresponding model given the wild-type backbone.

Table 1: Higher-order variant prioritization from low-order functional measurements across three FLIP2 datasets with continuous fitness scores. All @K metrics report $K = 10/100$; see Appendix C for metric definitions.

Dataset	Model	H_C@K ↑	EF@K ↑	NDCG@K ↑	Pearson ↑	Spearman ↑
Amylase	ProteinMPNN	1 / 13	2.97 / 3.86	0.71 / 0.67	0.449	0.388
	+SPPO	2 / 13	5.94 / 3.86	0.66 / 0.64	0.437	0.378
	+FPA	2 / 19	5.94 / 5.64	0.67 / 0.75	0.564	0.503
	+PIF w/o Neg	0 / 16	0 / 4.76	0.64 / 0.71	0.585	0.545
	+PIF w/o PFM	3 / 17	8.91 / 5.05	0.67 / 0.75	0.556	0.500
	+PIF (Full)	**4 / 24**	**11.89 / 7.13**	**0.73 / 0.78**	**0.648**	**0.599**
	ESM-IF	0 / 16	0 / 4.76	0.56 / 0.62	0.433	0.398
	+SPPO	0 / 16	0 / 4.76	0.51 / 0.64	0.461	0.429
	+FPA	2 / 20	5.95 / 5.95	0.64 / 0.72	0.574	0.557
	+PIF w/o Neg	**3** / 18	**8.92** / 5.35	0.78 / 0.74	0.585	0.567
	+PIF w/o PFM	**3 / 22**	**8.92 / 6.54**	0.59 / 0.72	0.570	0.556
	+PIF (Full)	**3** / 18	**8.92** / 5.35	**0.79 / 0.75**	**0.586**	**0.569**
IRED	ProteinMPNN	4 / 21	3.71 / 1.95	0.52 / 0.60	0.108	0.116
	+SPPO	1 / 16	0.92 / 1.48	0.44 / 0.59	0.106	0.115
	+FPA	2 / 30	1.85 / 2.78	0.60 / 0.65	0.175	0.182
	+PIF w/o Neg	5 / 37	4.64 / 3.43	0.63 / 0.67	0.184	0.187
	+PIF w/o PFM	3 / 25	2.78 / 2.32	0.61 / 0.64	0.170	0.176
	+PIF (Full)	**6 / 42**	**5.57 / 3.89**	**0.68 / 0.71**	**0.196**	**0.199**
	ESM-IF	4 / 22	3.71 / 2.04	0.60 / 0.63	0.125	0.137
	+SPPO	5 / 25	4.64 / 2.32	0.61 / 0.63	0.134	0.149
	+FPA	**7** / 34	**6.50** / 3.16	**0.68** / 0.68	0.160	0.180
	+PIF w/o Neg	5 / **39**	4.64 / **3.62**	0.66 / **0.70**	**0.167**	**0.188**
	+PIF w/o PFM	**7** / 32	**6.50** / 2.97	**0.68** / 0.67	0.152	0.169
	+PIF (Full)	**7** / 38	**6.50** / 3.53	0.67 / **0.70**	0.166	0.186
TrpB	ProteinMPNN	0 / 7	0 / 14.84	0.38 / 0.40	0.181	0.059
	+SPPO	1 / 5	21.20 / 10.60	0.30 / 0.36	0.268	0.305
	+FPA	0 / 13	0 / 27.55	0.37 / 0.47	**0.377**	0.432
	+PIF w/o Neg	1 / 11	21.20 / 23.31	0.38 / 0.45	0.360	**0.449**
	+PIF w/o PFM	0 / 16	0 / 33.91	0.39 / 0.49	**0.377**	0.433
	+PIF (Full)	**3 / 28**	**63.60 / 59.35**	**0.59 / 0.59**	0.359	0.433
	ESM-IF	4 / 22	84.79 / 46.63	0.44 / 0.50	0.243	0.119
	+SPPO	1/ 11	21.20 / 23.32	0.41 / 0.47	0.267	**0.448**
	+FPA	6 / 36	127.20 / 76.32	0.63 / 0.59	0.392	0.414
	+PIF w/o Neg	6 / 36	127.20 / 76.32	0.67 / 0.60	0.397	0.409
	+PIF w/o PFM	4 / 27	84.80 / 57.24	0.53 / 0.58	0.305	0.406
	+PIF (Full)	**8 / 48**	**169.60 / 101.76**	**0.70 / 0.66**	**0.404**	0.410

4.2 Evaluation

Higher-Order Variant Prioritization. Tables 1 and 2 show that PIF consistently improves higher-order variant prioritization for both ProteinMPNN and ESM-IF when evaluated on held-out sequences carrying three or more mutations, despite using experimentally measured supervision only from wild-type, single-, and double-mutant variants. This improvement is not confined to a single metric: it is reflected in the model's ability to retrieve truly active variants near the top of its ranking, enrich functional variants relative to chance, and better approximate the ideal ordering of the candidate pool. Taken together, these results support our central hypothesis that limited low-order measurements can guide prioritization across a substantially larger higher-order sequence space, rather than the learned preferences remaining confined to the narrow mutational neighborhood observed during training.

Ablations. Tables 1 and 2 further show that both major components of PIF contribute to the higher-order generalization. Removing the synthetic higher-order negatives (**w/o Neg**) generally weakens top-K prioritization, with particularly clear degradation on Amylase and NucB, indicating that these negatives provide useful supervision for transferring low-order functional preferences into the unseen

Table 2: Higher-order variant prioritization on the categorical NucB dataset. Since NucB provides discrete activity labels rather than continuous fitness scores, we define classes $\{2, 3\}$ as active for $H_C@K$ and EF@K. All @K metrics report $K = 10/100$; see Appendix C for metric definitions.

	ProteinMPNN			ESM-IF		
Method	$H_C@K \uparrow$	EF@$K \uparrow$	NDCG@$K \uparrow$	$H_C@K \uparrow$	EF@$K \uparrow$	NDCG@$K \uparrow$
Vanilla	5 / 47	2.27 / 2.13	0.46 / 0.41	3 / 29	1.36 / 1.32	0.21 / 0.24
+SPPO	0 / 10	0 / 0.45	0.07 / 0.15	1 / 21	0.45 / 0.95	0.19 / 0.29
+FPA	2 / 14	0.91 / 0.64	0.17 / 0.20	0 / 10	0 / 0.45	0.08 / 0.15
+PIF w/o Neg	1 / 8	0.45 / 0.36	0.22 / 0.16	0 / 12	0 / 0.54	0.07 / 0.16
+PIF w/o PFM	4 / 29	1.81 / 1.31	0.35 / 0.30	2 / 36	0.91 / 1.63	0.20 / 0.33
+PIF (Full)	**9 / 56**	**4.08 / 2.54**	**0.64 / 0.48**	**8 / 44**	**3.63 / 2.00**	**0.56 / 0.45**

Table 3: Sequence recovery (Seq. Rec. (%)) and perplexity (PPL) before and after PIF fine-tuning on four FLIP2 datasets. Results are evaluated on the single-chain (Single) and multi-chain (Multi) test sets of the respective pretrained models. Best value per architecture and column is marked in bold.

	Seq. Rec. (%) ↑		PPL ↓	
Model	Single	Multi	Single	Multi
ProteinMPNN	47.9	48.8	4.64	4.33
+PIF (Amylase)	47.7	48.6	4.56	4.25
+PIF (IRED)	48.5	49.3	4.45	4.16
+PIF (TrpB)	**49.3**	**50.1**	**4.33**	**4.07**
+PIF (NucB)	48.7	49.4	4.41	4.15

	Seq. Rec. (%) ↑		PPL ↓	
Model	Single	Multi	Single	Multi
ESM-IF	44.5	46.9	5.58	7.33
+PIF (Amylase)	44.1	46.5	5.64	6.87
+PIF (IRED)	44.4	47.0	5.57	7.06
+PIF (TrpB)	**45.0**	**48.0**	5.42	**4.93**
+PIF (NucB)	44.9	47.8	**5.35**	6.09

higher-order sequence space. Removing protein-foundation-model conditioning (**w/o PFM**) also reduces performance across most datasets, showing that sequence-informed representations provide complementary information beyond the preference objective itself. Although individual metrics occasionally favor an ablated variant, the full PIF generally achieves the strongest HC@K, EF@K, and NDCG@K results across both ProteinMPNN and ESM-IF backbones. Together, these results suggest that cross-order supervision and foundation-model conditioning provide complementary benefits for higher-order variant prioritization, beyond those obtained from preference fine-tuning alone.

Functional Alignment Preserves Inverse-Folding Capability. Table 3 reports sequence recovery and perplexity on the underlying structural design task, complementing the higher-order prioritization results in Section 4.2. To reflect each architecture's native evaluation protocol, we report results on each model's original held-out test split. The ProteinMPNN evaluation comprises 2,636 single-chain and 4,773 multi-chain structures (7,409 total), whereas the ESM-IF evaluation comprises 772 single-chain and 1,035 multi-chain structures (1,807 total). Aligned models remain close to, and in several cases improve upon, their pretrained counterparts, indicating that functional guidance does not come at the cost of the base structure-conditioned sequence prior.

Generalization Trend with Increasing Training Mutational Load. Figure 1 examines extrapolation on Amylase as the maximum training mutational load increases from $k_{\text{train}} = 1$ to 4. Across both ProteinMPNN and ESM-IF, expanding the training regime generally improves performance on higher-order test variants. Importantly, much of the gain is already realized by $k_{\text{train}} = 2$, while incorporating variants beyond double mutants yields comparatively smaller and less consistent improvements. This diminishing return highlights the practical value of the $k_{\text{train}} = 2$ setting in Table 1: low-order measurements capture most of the extrapolation benefit, while avoiding the rapidly increasing experimental burden of characterizing higher-order variants.

Discrimination Between Functional Classes. Figure 2 reports ROC curves comparing pretrained ProteinMPNN and ESM-IF backbones against their +PIF (Full) counterparts across all four datasets. AUROC measures how consistently the model assigns higher scores to positive samples than to

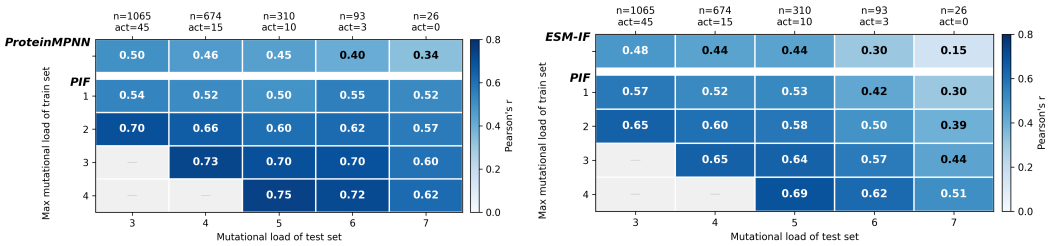


Figure 1: Fine-grained extrapolation on Amylase across training and test mutational loads, measured by Pearson correlation, for ProteinMPNN (**Left**) and ESM-IF (**Right**). Columns denote test mutational load and rows the maximum mutational load used for PIF fine-tuning. The Baseline row is the vanilla pretrained model; cells with $k_{\text{test}} \leq k_{\text{train}}$ are omitted. Column headers report the number of test variants (n) and active variants exceeding wild-type activity (act). Corresponding Spearman and H_C@10 trends are reported in Figure 3.

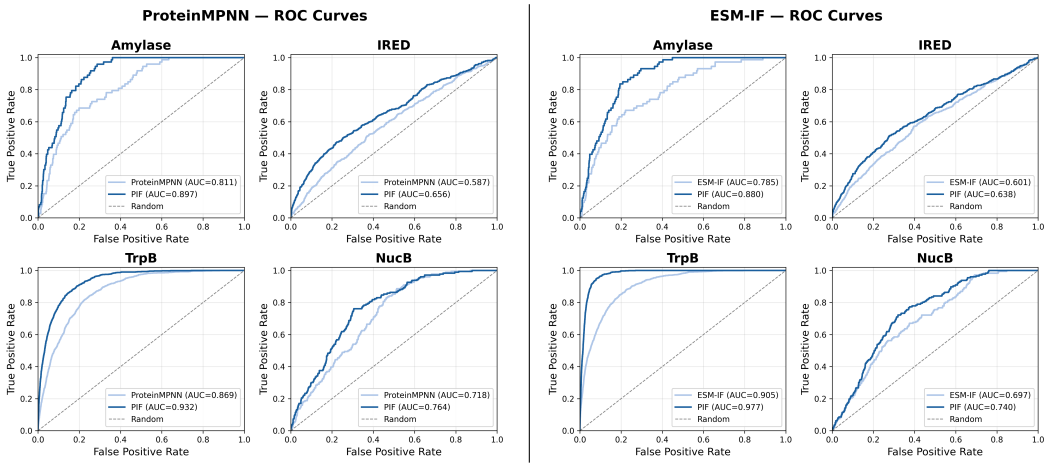


Figure 2: ROC curves for discriminating positive (high-activity) from negative (low/no-activity) sequences by model score, comparing each backbone with its +PIF (Full) fine-tuned counterpart across all four datasets, for ProteinMPNN (**Left**) and ESM-IF (**Right**). AUROC equals the pairwise accuracy $P(\hat{y}(x^+) > \hat{y}(x^-))$, computed exhaustively over all positive–negative test sequence pairs.

negative samples. Across all datasets and both architectures, +PIF (Full) consistently achieves higher AUROC than the pretrained backbone, showing that preference alignment improves the ranking of functionally active sequences above inactive ones–directly relevant to candidate prioritization.

5 Conclusion

We studied whether limited low-order functional measurements can guide the prioritization of unseen higher-order protein variants, combining cross-order preference construction, critical-site alignment, and foundation-model-conditioned representations to transfer supervision from single and double mutants to variants with three or more mutations. Our evaluation spans FLIP2 datasets with continuous fitness scores and categorical activity labels, while the alignment framework is agnostic to the assay or readout and can accommodate any experimentally accessible continuous or discrete functional measurement. Across datasets and two inverse-folding architectures, aligned models improve higher-order variant prioritization while largely preserving sequence-design capability, showing that experimentally accessible low-order measurements can guide navigation of much larger combinatorial protein design spaces regardless of the specific functional property being engineered.

AI USE STATEMENT

Generative AI tools were used to assist with editing and polishing the writing in this manuscript, including improving clarity, grammar, and phrasing of drafted text. AI tools were not used to design experiments, generate results, or produce the core methodological ideas presented in this work. The authors take full responsibility for the final content, experimental results, and conclusions.

ETHICS STATEMENT

This work develops computational methods for protein inverse folding and higher-order variant prioritization using experimentally measured low-order mutants. The study is methodological and does not involve human participants, personal data, or animal experiments. All datasets used (FLIP2 Amylase, IRED, TrpB, and NucB) are publicly available and were collected in prior published work. While improved protein design methods may support beneficial applications in biotechnology and biomedicine, they may also have broader dual-use implications. Our experiments are conducted on established protein datasets and focus on benchmarking sequence prioritization rather than designing pathogenic, toxic, or otherwise harmful biological functions. We encourage responsible use of the proposed methods in accordance with applicable biosafety, biosecurity, and experimental oversight practices.

REPRODUCIBILITY STATEMENT

Implementation and experimental details are provided in the Appendix: Model architecture and FiLM conditioning (Appendix B), optimization hyperparameters (Appendix B.4), dataset sources and split construction for all four datasets (Appendix A), and formal definitions of all evaluation metrics (Appendix C). Dataset splits are constructed deterministically from publicly available FLIP2 data using fixed random seeds (Appendix A). Code to reproduce the training and evaluation pipeline will be released upon publication.

REFERENCES

Justas Dauparas, Ivan Anishchenko, Nathaniel Bennett, Hua Bai, Robert J Ragotte, Lukas F Milles, Basile IM Wicky, Alexis Courbet, Rob J de Haas, Neville Bethel, et al. Robust deep learning–based protein sequence design using proteinmpnn. *Science*, 378(6615):49–56, 2022.

Justas Dauparas, Gyu Rie Lee, Robert Pecoraro, Linna An, Ivan Anishchenko, Cameron Glasscock, and David Baker. Atomic context-conditioned protein sequence design using ligandmpnn. *Nature Methods*, 22(4):717–723, 2025.

Kieran Didi, Sarah Alamdari, Alex Xijie Lu, Bruce James Wittmann, Kadina E Johnston, Ava P Amini, Ali Madani, Maya Czeneszew, Christian Dallago, and Kevin K Yang. FLIP2: Expanding protein fitness landscape benchmarks for real-world machine learning applications. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Ue8aOgz9T9.

Maximilian Gantz, Simon V. Mathis, Friederike E. H. Nintzel, Matthew Penner, Oskar James Klein, Anne-Cathrin Prowald, Pietro Liò, and Florian Hollfelder. Microdroplet screening rapidly profiles a biocatalyst to enable its ai-assisted engineering. *bioRxiv*, 2024. doi: 10.1101/2024.04.08.588565.

Chloe Hsu, Robert Verkuil, Jason Liu, Zeming Lin, Brian Hie, Tom Sercu, Adam Lerer, and Alexander Rives. Learning inverse folding from millions of predicted structures. In *International Conference on Machine Learning*, pages 8946–8970. PMLR, 2022.

John B Ingraham, Max Baranov, Zak Costello, Karl W Barber, Wujie Wang, Ahmed Ismail, Vincent Frappier, Dana M Lord, Christopher Ng-Thow-Hing, Erik R Van Vlack, et al. Illuminating protein space with a programmable generative model. *Nature*, 623(7989):1070–1078, 2023.

Vijay Jayaraman, Saacnicteh Toledo-Patiño, Lianet Noda-García, and Paola Laurino. Mechanisms of protein evolution. *Protein Science*, 31(7):e4362, 2022.

Kadina E. Johnston, Patrick J. Almhjell, Ella J. Watkins-Dulaney, Grace Liu, Nicholas J. Porter, Jason Yang, and Frances H. Arnold. A combinatorially complete epistatic fitness landscape in an enzyme active site. *Proceedings of the National Academy of Sciences*, 121(32):e2400439121, 2024. doi: 10.1073/pnas.2400439121.

Michal Kmicikiewicz, Vincent Fortuin, and Ewa Szczurek. Prospero: Active learning for robust protein design beyond wild-type neighborhoods. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=wSDE3karoF.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Nilufer Tamatgar, Soobin Park, Yinghua Yao, Xixian Chen, and Yuangang Pan. From structure to function: Preference alignment for function-aware protein inverse folding. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, pages 12066–12077, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400722592. doi: 10.1145/3770855.3818896. URL https://doi.org/10.1145/3770855.3818896.

Neil Thomas et al. Engineering highly active nuclease enzymes with machine learning and high-throughput screening. *Cell Systems*, 2025. doi: 10.1016/j.cels.2025.02.005.

Vincent Q. Tran, Matthew Nemeth, Liam J. Bartie, Sita S. Chandrasekaran, Alison Fanton, Hyungseok C. Moon, Brian L. Hie, Silvana Konermann, and Patrick D. Hsu. Rapid directed evolution guided by protein language models and epistatic interactions. *Science*, 392(6798): eaea1820, 2026. doi: 10.1126/science.aea1820.

Floris van der Flier, Dave Estell, Sina Pricelius, Lydia Dankmeyer, Sander van Stigt Thans, Harm Mulder, Rei Otsuka, Frits Goedegebuur, Laurens Lammerts, Diego Staphorst, Aalt D. J. van Dijk, Dick de Ridder, and Henning Redestig. Enzyme structure correlates with variant effect predictability. *Computational and Structural Biotechnology Journal*, 23:3489–3497, 2024. doi: 10.1016/j.csbj.2024.09.007.

Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

Talal Widatalla, Rafael Rafailov, and Brian L. Hie. Aligning protein-generative models to experimental fitness with ProteinDPO. *Nature Methods*, 2026. doi: 10.1038/s41592-026-03137-3.

Nicholas C Wu, Lei Dai, C Anders Olson, James O Lloyd-Smith, and Ren Sun. Adaptation in protein fitness landscapes is facilitated by indirect paths. *elife*, 5:e16965, 2016.

Yue Wu, Zhiqing Sun, Rina Hughes, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. In *International Conference on Learning Representations*, volume 2025, pages 91558–91582, 2025a.

Yue Wu, Zhiqing Sun, Huizhuo Yuan, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=a3PmRgAB5T.

Zachary Wu, SB Jennifer Kan, Russell D Lewis, Bruce J Wittmann, and Frances H Arnold. Machine learning-assisted directed protein evolution with combinatorial libraries. *Proceedings of the National Academy of Sciences*, 116(18):8852–8858, 2019.

Kevin K Yang, Zachary Wu, and Frances H Arnold. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694, 2019.

A Dataset Details

We use four datasets from the FLIP2 benchmark (Didi et al., 2026), a protein fitness benchmark comprising seven new datasets covering diverse protein engineering tasks, including enzyme activity, protein–protein interactions, and light-sensitive proteins. FLIP2 extends the original FLIP benchmark, which focused on thermostability, binding, and viral capsid viability, and introduces evaluation splits designed to assess generalization under settings relevant to real-world protein engineering campaigns.

A.1 Split Construction Methodology

FLIP2 provides a native *two-to-many* number split for three of the four datasets used in this work (IRED, TrpB and NucB): variants with 0, 1, or 2 mutations relative to the wild type form the training pool, and all variants with more than two mutations are held out for testing. For these two datasets, we use the FLIP2-provided split directly, without modification.

Amylase, however, is distributed by FLIP2 only with a *one-to-many* split (train on single mutants only; test on everything else), which does not match the setting of interested studied (Section 3.1). We therefore construct a *two-to-many* split for Amylase from the underlying data, following the same logic as FLIP2's native two-to-many splits for the other datasets:

- For each sequence, the number of mutations relative to the (trimmed) wild-type sequence is computed as the Hamming distance between the mutant and wild-type sequences.
- Sequences with more than two mutations are assigned to the **test** set.
- Sequences with one or two mutations form the pool used for training and validation. This pool is split into **train** (85%) and **validation** (15%) subsets using a stratified split on the number of mutations (`random_state=42`), preserving the mutation-count distribution across both subsets.
- The wild-type sequence itself is additionally included in the training set.

A.2 Dataset Split Statistics

Table 4 summarizes the total number of sequences per split across all four datasets, following the split construction described in Appendix A.1. Per-dataset breakdowns by mutation count, including the subset of higher-activity sequences at each mutation count, are given in Tables 5, 6, and 7 in the respective dataset subsections below.

Table 4: Number of sequences per split across the FLIP2 datasets used in this work. The training set for each dataset additionally includes one wild-type reference sequence.

Split	Amylase	IRED	TrpB	NucB
Train	1,304	3,745	8,632	11,039
Validation	231	662	2,158	1,949
Test	2,170	4,178	217,507	41,831

A.3 AMYLASE

The wild-type (WT) reference sequence and corresponding PDB structure (1UA7) are derived from van der Flier et al. (2024) and were obtained through the dataset's accompanying GitHub repository[1]. The provided WT sequence of Amylase is:

LTAPSIKSGTILHAWNWSFNTLKHNMKDIHDAGYTAIQTSPINQVKEGNQGDKSMSNWY
WLYQPTSYQIGNRYLGTEQEFKEMCAAAEEYGIKVIVDAVINHTTSDYAAISNEVKSIPNW
THGNTPIKNWSDRWDVTQNSLSGLYDWNTQNTQVQSYLKRFLDRALNDGADGFRFDAA
KHIELPDDGSYGSQFWPNITNTSAEFQYGEILQDSVSRDAAYANYMDVTASNYGHSIRSAL
KNRNLGVSNISHYAVDVSADKLVTWVESHDTYANDDEESTWMSDDDIRLGWAVIASRS
GSTPLFFSRPEGGGNGVRFPGKSQIGDRGSALFEDQAITAVNRFHNVMAGQPEELSNPNGN
NQIFMNQRGSHGVVLANAGSSSVSINTATKLPDGRYDNKAGAGSFQVNDGKLTGTINARS
VAVLYPD

As the first three residues of the wild-type sequence are not included in the backbone structure (PDB: 1UA7), they are removed from all sequences (including the wild-type) prior to processing, so that sequence positions align with the structural model.

Table 5: Amylase: number of sequences per mutation count across splits. Values in parentheses indicate the subset with activity score greater than the wild-type activity score.

Split	# Mutations	# Sequences (higher activity)
Train	1	415 (36)
	2	889 (43)
Validation	1	73 (5)
	2	158 (8)
Test	3	1065 (45)
	4	674 (15)
	5	310 (10)
	6	93 (3)
	7	26 (0)
	8	2 (0)

A.4 IRED

The Imine Reductase (IRED) dataset contains activity measurements from a microfluidic screen of *Streptosporangium roseum* imine reductase variants from an error-prone PCR library, comprising 17,143 total variants (1,207 single mutants), including variants with up to 15 mutations (Gantz et al., 2024). FLIP2 provides a native *two-to-many* split for this dataset (train on 0, 1, or 2 mutations; test on all others), which we use directly (Appendix A.1). The WT reference sequence and used structure were adopted from the original study which generated this dataset (PDB: 50CM, Chain B) (Gantz et al., 2024) : The provided WT sequence of IRED is :

MRDTDVTVLGLGLMGQALAGAFLKDGHATTVWNRSEGKAGQLAEQGAVLASSARDAA
EASPLVVVCVSDHAAVRAVLDPLGDVLAGRVLVNLTSGTSEQARATAEWAAERGITYLD
GAIMAIPQVVGTADAFLLYSGPEAAYEAHEPTLRSLGAGTTYLGADHGLSSLYDVALLGI
MWGTLNSFLHGAALLGTAKVEATTFAPFANRWIEAVTGFVSAYAGQVDQGAYPALDATI
DTHVATVDHLIHESEAAGVNTELPRLVRTLADRALAGGQGGLGYAAMIEQFRSPS

A.5 TRPB

The Tryptophan Synthase β-Subunit (TrpB) dataset contains growth-based fitness measurements for combinatorially complete landscapes across multiple sets of interacting residues, comprising 228,298 total variants across 10 sub-landscapes spanning 20 positions and up to 4 combinatorical mutations (Johnston et al., 2024). FLIP2 provides both a *one-to-many* and a native *two-to-many* split for this

[1] https://github.com/florisvdf/mutation-predictability/tree/main

Table 6: IRED: number of sequences per mutation count across splits. Values in parentheses indicate the subset with activity score greater than the wild-type activity score.

Split	# Mutations	# Sequences (higher activity)
Train	1	1037 (191)
	2	2708 (475)
Validation	1	170 (41)
	2	492 (80)
Test	3	2053 (250)
	4	1172 (123)
	5	569 (49)
	6	236 (22)
	7	89 (4)
	8	41 (1)
	9	10 (0)
	10	4 (0)
	11	2 (1)
	12	1 (0)
	13	/
	14	/
	15	1 (0)

dataset; we use the latter directly (Appendix A.1). Alike to the analysis of the original publication (Johnston et al., 2024) we used the identical reference structure and WT sequence (PDB: 8VHH, Chain A):

MKGYFGPYGGQYVPEILMGALEELEAAYEGIMKDESFWKEFNDLLRDYAGRPTPLYFAR
RLSEKYGARVYLKREDLLHTGAHKINNAIGQVLLAKLMGKTRIIAETGAGQHGVATATA
AALFGMECVIYMGEEDTIRQKLNVERMKLLGAKVVPVKSGSRTLKDAIDEALRDWITNL
QTTYYVFGSVVGPHPYPIIVRNFQKVIGEETKKQIPEKEGRLPDYIVACVSGGSNAAGIFYP
FIDSGVKLIGVEAGGEGLETGKHAASLLKGKIGYLHGSKTFVLQDDWGQVQVSHSVSAGL
DYSGVGPEHAYWRETGKVLYDAVTDEEALDAFIELSRLEGIIPALESSHALAYLKKINIKG
KVVVVNLSGRGDKDLESVLNHPYVRERIRLEHHHHHH

To match the sequences to the provided backbone structure (PDB: 8VHH, Chain A), the first residue was excluded from all provided mutational sequences and WT sequence. Additionally, the last 8 residues were excluded from the WT sequence and the last residue from the backbone structure to be coherent in length and amino acids with the provided mutational sequences from the FLIP2 dataset.

Table 7: TrpB: number of sequences per mutation count across splits. Values in parentheses indicate the subset with activity score greater than the wild-type activity score.

Split	# Mutations	# Sequences (higher activity)
Train	1	316 (8)
	2	8316 (92)
Validation	1	63 (6)
	2	2095 (25)
Test	3	87989 (496)
	4	129518 (530)

A.6 NucB

Endonucleases such as *Bacillus licheniformis* Nuclease B (NucB) degrade DNA and have potential applications in chronic wound care by degrading extracellular DNA required for biofilm formation. This dataset measures nuclease activity at pH 7 for variants from an error-prone PCR library, com-

prising 55,760 total variants (Thomas et al., 2025). Unlike the other three datasets, NucB provides discrete activity categories rather than continuous fitness scores. We therefore evaluate it using retrieval metrics applicable to categorical activity labels, including H_C@K, EF@K, and NDCG@K, with classes $\{2, 3\}$ treated as active for H_C@K and EF@K (Section 4.2). FLIP2 provides a native *two-to-many* split for this dataset, which we use directly (Appendix A.1).

The wild-type (WT) reference sequence and corresponding PDB structure (5OMT) are derived from Thomas et al. (2025) and were obtained through the dataset's accompanying GitHub repository[2] and RCSB entry. The provided WT sequence of NucB is:

MIKKWAVHLLFSALVLLGLSGGAAYSPQHAEGAARYDDVLYFPASRYPETGAHISDAIKA
GHADVCTIERSGADKRRQESLKGIPTKPGFDRDEWPMAMCEEGGKGASVRYVSSSDNRG
AGSWVGNRLNGYADGTRILFIVQ

As the reference backbone structure (PDB: 5OMT) does not contain the first 33 residues of the provided sequences (mutational and WT sequences), they were removed from all sequences to map them to the provided backbone structure. As the truncation spans 33 residues, it created identical sequences (containing mutations only in the first 33 residues) with different activity bins. Thus, duplication sequences were removed, retaining only the first occurrence.

Table 8: NucB: number of sequences per mutation count across splits.

Split	# Mutations	# Categories 0,1,2,3
Train	1	1281,450,35,/
	2	5822,2480,962,8
Validation	1	202,74,5,/
	2	1052,444,171,1
Test	3	5528,2272,1890,29
	4	4550,1755,1801,63
	5	3519,1293,1409,31
	6	2816,788,873,18
	7	2059,359,448,5
	8	1519,182,392,2
	9	758,115,385,4
	10	1413,222,854,10
	11	1520,187,650,10
	12	814,83,267,4
	13	221,26,28,/
	14	109,6,16,/
	15	205,11,29,/
	16	62,1,3,/
	17	18,2,1,/
	18	5,/,/,/
	19	47,1,/,/
	20	112,/,/,/
	21	25,1,/,/
	22	4,/,/,/
	23	1,/,/,/

A.7 Inactive Sequence Set

To support the inactive recombination setting, a set of low-activity ("inactive") sequences is derived from the training split. For Amylase and TrpB, sequences are ranked by their activity score in ascending order, and the bottom 10% (quantile cutoff of 0.1) are selected as inactive sequences; for reference, the wild-type activity score is 0.155 for Amylase and 1.0 for TrpB, while the 10th-percentile activity score of the training set is 0.009 and -0.065, respectively. For IRED, the original

[2] https://github.com/florisvdf/mutation-predictability/tree/main

publication provides experimentally determined inactive sequences directly, removing the need to construct an in-silico low-activity set. Table 9 summarizes the active/inactive threshold used for each dataset.

In the categorical setting of NucB, category 0 describes non-functional sequences, equivalent to our definition of inactivity; all sequences categorized as 0 in the training set are therefore used as the inactive set.

A.8 Inactive Recombination via Mutation Combination

To generate additional inactive-by-construction sequences beyond the quantile-based inactive set described above, we construct new recombinant sequences by combining mutations associated with inactivity. This procedure targets positions and substitutions that are characteristic of low-activity variants, and applies them either to existing high-activity training sequences or directly to the wild-type sequence.

A.8.1 Mutation Pool Construction

Starting from the training split and the inactive sequence set (Appendix A.7), we identify the set of unique single-position mutations (relative to the wild type) observed in each: *active mutations*, drawn from all training sequences, and *inactive mutations*, drawn from the inactive sequence set. Inactive mutations that also occur among the active mutations are discarded, isolating mutations uniquely associated with inactive sequences; if this filtering removes all candidates, the unfiltered inactive mutation set is used instead. The 25 most frequently occurring filtered inactive mutations form the mutation pool used for recombination.

A.8.2 Recombination Strategies

Two complementary strategies generate new inactive-by-construction sequences by applying combinations of mutations from the pool:

Active-sequence-based recombination. Base sequences are training sequences with an activity score above a dataset-specific threshold (Table 9). For each such sequence and each combination size $r \in \{3, 4, 5\}$, combinations of r mutations are drawn from the pool, restricted to combinations with distinct positions that do not overlap with positions already mutated in the base sequence. Up to 3 valid combinations per (sequence, r) pair are applied, producing recombinant sequences that layer newly added inactive-associated mutations on top of an already-active mutational background.

Table 9: Threshold of active and inactive subsets for recombination per dataset.

Dataset	Active sequences	Inactive sequences
Amylase	top 15%	bottom 10%
IRED	≥ 0.5	exp. derived
TrpB	top 5%	bottom 10%
NucB	class ≥ 2	class $= 0$

Wild-type-based recombination. Base sequence is the wild type. For each combination size $r \in \{3, 4, 5\}$, combinations of r mutations with distinct positions are drawn from the pool, up to 600 combinations per value of r and applied directly to the wild-type sequence. This produces recombinant sequences built entirely from inactive-associated mutations on an otherwise unmutated background.

A.8.3 Output

The two strategies are saved separately for each dataset; Table 10 reports the resulting number of active-sequence-based and wild-type-based recombinants per dataset.

Table 10: Number of recombinant sequences per setting and dataset.

Dataset	# Active-sequence recombinants	# WT recombinants
Amylase	1,764	1,800
IRED	810	1,800
TrpB	2,592	629
NucB	9,043	1,800

B Training Details

B.1 Architecture and Implementation Details

We fine-tune pretrained IF base models (ProteinMPNN and ESM-IF) using a shared conditioning mechanism. Each base model follows an encoder–decoder architecture, in which a structural encoder produces per-residue representations from the input backbone, and a decoder subsequently predicts residue identities conditioned on these representations. To incorporate protein language model information, we augment each base model with an ESM-conditioned FiLM (Feature-wise Linear Modulation) module, inserted immediately prior to the decoder, modulating the encoder's output representations before they are passed downstream. The strength of this conditioning is controlled by a single learnable scalar gate parameter g, described in detail below.

ESM2 conditioning embeddings. The conditioning input to the FiLM module is a per-residue embedding from ESM2 (Lin et al., 2023), computed once per sequence and held fixed. We use the 650M-parameter ESM2 checkpoint (33 transformer layers, 1280-dim output representations), loaded via `esm.pretrained.esm2_t33_650M_UR50D()` in the standard case, or from a local checkpoint when one is supplied. For a sequence of length L, we extract the final-layer (layer 33) per-residue representations (`repr_layers=[33]`), giving a $[L, 1280]$ embedding tensor that is aligned position-for-position with the structural encoder's node embeddings before being passed into the FiLM module's projection. The ESM2 model is run in evaluation mode throughout, and its embeddings are precomputed offline for all sequences in the training, validation, and test pools, avoiding a forward pass through ESM2 at every training step.

FiLM module. The FiLM parameters γ_i, β_i are produced from the per-residue ESM2 embedding $e_i \in \mathbb{R}^{1280}$ by a lightweight projection module, $f_{\text{film}} : \mathbb{R}^{1280} \to \mathbb{R}^{2H}$, where H is the model's hidden dimension. Each residue's embedding is projected independently:

$$u_i = \mathrm{LayerNorm}(e_i), \tag{9}$$

$$z_i = \mathrm{Dropout}\big(\mathrm{GELU}(W_1 u_i + b_1)\big), \qquad W_1 \in \mathbb{R}^{H \times 1280}, \tag{10}$$

$$[\gamma_i\,;\,\beta_i] = W_2 z_i + b_2, \qquad W_2 \in \mathbb{R}^{2H \times H}, \tag{11}$$

where $[\gamma_i\,;\,\beta_i] \in \mathbb{R}^{2H}$ denotes the concatenation of γ_i and β_i, so that $\gamma_i, \beta_i \in \mathbb{R}^H$ are simply the first and second halves of $W_2 z_i + b_2$. Since u_i, z_i depend only on e_i, this projection is applied position-wise: γ_i, β_i depend only on residue i's own ESM2 embedding, with no mixing across positions.

FiLM insertion point. In ProteinMPNN, the encoder is a stack of message-passing layers operating over a k-nearest-neighbor backbone graph, producing a per-residue node embedding $h_i \in \mathbb{R}^d$ for each position i after the final encoder layer; these embeddings are ordinarily passed directly into the autoregressive decoder, which predicts residue identities conditioned on h_i, the residue graph, and previously decoded residues. We intercept this hand-off: immediately after the last encoder layer and before any decoder layer consumes h_i, we apply

$$h'_i = h_i \odot \big(1 + \sigma(g)\,\tanh(\gamma_i)\big) + \sigma(g)\,\beta_i, \tag{12}$$

where (γ_i, β_i) are produced per-residue by the FiLM projection module (below) and $\sigma(g)$ is the sigmoid of the single scalar gate. As $\sigma(g) \to 0$, $h'_i \to h_i$, recovering the unmodulated base model; as $\sigma(g) \to 1$, the full FiLM modulation is applied. The gate is initialized at $g = 0$, yielding

Table 11: Dataset-specific pair-construction thresholds. For continuous-target datasets, an on-the-fly pair is retained only if the activity difference between its two members exceeds τ_{pair}.

Dataset	τ_{pair}
Amylase	0.1
TrpB	0.1
IRED	1.0
NucB	n/a (class-based)

$\sigma(g) = 0.5$ and giving the FiLM pathway moderate initial influence rather than starting fully active or fully suppressed. We bound γ_i with $\tanh$ to prevent the multiplicative term from producing large-magnitude rescalings early in training, and zero out γ_i, β_i at padded positions using the sequence mask before they are applied, so padding never contributes spurious modulation. Decoding then proceeds exactly as in the unmodified base model, but on h'_i in place of h_i. In ESM-IF, the encoder is a GVP-Transformer that similarly outputs per-residue backbone-conditioned embeddings prior to the autoregressive decoder; we apply the identical FiLM operation at the analogous encoder-output boundary. In both architectures, the FiLM projection parameters and the scalar gate g are the only new trainable parameters introduced; the surrounding encoder and decoder layers, and all other hyperparameters of the base model, are left unchanged.

Gate. The scalar gate $g \in \mathbb{R}$ is a single free parameter, shared across all residue positions i, all batch elements, and – since FiLM is applied only once, immediately after the final encoder layer – all layers. It is not a function of e_i or of position; it is the model's single global, learnable dial on the overall strength of ESM2 conditioning, independent of any per-position or per-layer variation in γ_i, β_i themselves. Because g is a single scalar with a qualitatively different role from the rest of the model's parameters, we optimize it separately, using a learning rate two to three orders of magnitude higher than the base learning rate used for all other parameters (Appendix B.4), allowing the gate to move quickly to an appropriate operating point early in training rather than being constrained to the slower schedule needed for the much larger set of pretrained weights.

B.2 Preference Pair Construction

Training pairs are drawn from two sources, combined and shuffled at each training step:

- **On-the-fly pairs:** constructed within a freshly sampled minibatch of labeled training sequences at each step. For datasets with continuous activity scores (Amylase, TrpB, IRED), a pair is retained only if the activity difference between its two members exceeds a dataset-specific threshold τ_{pair} (Table 11).

  For NucB, whose activity labels are ordinal classes, pairs are instead formed between sequences belonging to distinct classes, with the higher class taken as the preferred member.
- **Cross-order pairs:** sampled independently at each step from two fixed pools. The positive pool is defined by a top quantile of the training activity distribution for Amylase and TrpB, and by an activity threshold for IRED and NucB(Table 9). The negative pool consists of recombinant inactive-by-construction sequences generated by both the active-sequence-based and wild-type-based procedures described in Appendix A.8.

B.3 Training Objective

The model is optimized with a combined objective consisting of a pairwise preference loss over sequence pairs from each fitness dataset and a cross-entropy regularization term computed on a separate, general-purpose structure corpus comprising structures common to both ProteinMPNN's and ESM-IF's original pretraining data, preserving the model's baseline sequence-design capability during task-specific fine-tuning:

$$\mathcal{L} = \mathcal{L}_{\text{CPO}} + \lambda_{\text{CE}}\,\mathcal{L}_{\text{CE}} + \mathbb{I}_{\text{cat}}\,\lambda_{\text{cls}}\,\mathcal{L}_{\text{cls}}, \qquad \lambda_{\text{CE}} = 0.2, \tag{13}$$

where $\mathbb{I}_{\text{cat}} = 1$ for categorical datasets (NucB) and 0 otherwise. $\mathcal{L}_{\text{CE}}$ denotes the standard per-residue masked negative log-likelihood loss used in ProteinMPNN training. The preference loss

$\mathcal{L}_{\mathrm{CPO}}$, following Self-Play Preference Optimization (Wu et al., 2025a), is computed per sequence pair as described in Algorithm 1 and averaged over all pairs in the batch.

Our implementation directly instantiates Algorithm 1. For a preference pair (x^+, x^-) with per-residue log-probabilities $\log p$ under the policy and frozen reference model, we compute the differing-position mask

$$d = (d_1, \ldots, d_L), \qquad d_i = \begin{cases} 1 & x_i^+ \neq x_i^- \\ 0 & x_i^+ = x_i^- \end{cases}, \tag{14}$$

combine it with the sequence padding mask, and average the masked log-probabilities over positions in d (clamping the mask sum to a minimum of 1 to avoid division by zero for pairs with no differing positions in a batch). The resulting per-sequence scaled log-ratios $a = \beta(\bar{\ell}_\theta^F(x^+) - \bar{\ell}_{\mathrm{ref}}^F(x^+))$ and $b = \beta(\bar{\ell}_\theta^F(x^-) - \bar{\ell}_{\mathrm{ref}}^F(x^-))$ give

$$\mathcal{L}_{\mathrm{CPO}} = \mathbb{E}\big[(a - \tfrac{1}{2})^2 + (b + \tfrac{1}{2})^2\big], \tag{15}$$

averaged over all pairs in the batch. The regularization term $\mathcal{L}_{\mathrm{CE}}$ is the standard masked per-residue negative log-likelihood, normalized per sequence by the number of unmasked residues before averaging over the batch, identical to the cross-entropy loss used to train the base ProteinMPNN/ESM-IF models.

For categorical datasets, the classification head g_{cls} is trained on labeled low-order samples $\mathcal{D}_{\mathrm{low}}^{\mathrm{cat}}$ only, since synthetic higher-order negatives lack experimentally measured category labels:

$$\mathcal{L}_{\mathrm{cls}} = \mathbb{E}_{(x,c)\sim\mathcal{D}_{\mathrm{low}}^{\mathrm{cat}}}\left[\mathrm{CE}\big(g_{\mathrm{cls}}(\bar{h}(x)),\, c\big)\right], \tag{16}$$

where $\bar{h}(x) = \frac{\sum_i m_i h_i}{\sum_i m_i}$ is the mean-pooled decoder representation over valid residues $m_i \in \{0, 1\}$, and c is the discrete activity category label.

Algorithm 1 Precision Inverse Folding (PIF)

Require: Preference dataset $\mathcal{D}$, categorical flag $\mathbb{I}_{\mathrm{cat}}$, reference model π_{ref}, scale β, trade-off coefficient λ_{CE} and λ_{cls}, critical-site mask $\mathcal{F}$, learning rate η, epochs T.
Ensure: Function-aware inverse folding model π_θ.

Initialize $\pi_\theta \leftarrow \pi_{\mathrm{ref}}$.
for epoch $t = 1, 2, \ldots, T$ **do**
  for mini-batch $\mathcal{D}_{\mathrm{sub}} \subseteq \mathcal{D}$ **do**
    Compute $\mathcal{L}_{\mathrm{CPO}} = \sum_{(y^+,y^-,x)\in\mathcal{D}_{\mathrm{sub}}} \ell_{\mathrm{CPO}}$.
    Compute $\mathcal{L}_{\mathrm{CE}} = \sum_{(y^+,y^-,x)\in\mathcal{D}_{\mathrm{sub}}} \ell_{\mathrm{CE}}$.
    $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{CPO}} + \lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \mathbb{I}_{\mathrm{cat}}\lambda_{\mathrm{cls}}\mathcal{L}_{\mathrm{cls}}$. See equation 8 for more details.
    $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}$.
  end for
end for

B.4 Optimization

Training was performed for a fixed number of epochs and the best epoch based on the acuurcay of the validation set was chosen. We collect the non-default hyperparameters in Table 12.

C Evaluation Metric Definitions

Let $\mathcal{D}_{\mathrm{test}}$ denote the test pool for a given dataset, with $n = |\mathcal{D}_{\mathrm{test}}|$. For each sequence $x \in \mathcal{D}_{\mathrm{test}}$, let $y(x)$ denote its experimentally measured activity score and $\hat{y}(x)$ the model's score, and let $\mathrm{Top}_K(\hat{y})$ denote the set of K sequences ranked highest by the model. We call a sequence *active* if $y(x) > \tau_{\mathrm{WT}}$, where τ_{WT} is the measured wild-type activity for that dataset.

$$\mathcal{A} = \{\, x \in \mathcal{D}_{\mathrm{test}} : y(x) > \tau_{\mathrm{WT}} \,\}, \qquad \rho = \frac{|\mathcal{A}|}{n}, \tag{17}$$

so that ρ is the fraction of the pool that would be found active by random selection.

Table 12: Training hyperparameters

Hyperparameter	**ProteinMPNN**	**ESM-IF**
scaling factor β	0.2	0.2
recovery loss λ_{CE}	0.2	0.2
classification loss λ_{cls}	1	1
Optimizer	AdamW	AdamW
Optimizer params	β_1=0.9, β_2=0.98, ϵ=10^{-9}	β_1=0.9, β_2=0.98, ϵ=10^{-8}
Weight decay	0.01	0.1
Learning rate	1×10^{-5}	1×10^{-6}
Gate learning rate	5×10^{-3}	5×10^{-3}
Epochs	50	50
Batch size	16	10

C.1 Hit Count@K (H_C@K)

We rank all test sequences by model score and take the top-K, then count how many of these selected sequences exceed wild-type activity:

$$\text{HC@}K = \left|\{i \in \text{Top}_K(\hat{y}) : y_i > \tau_{\text{WT}}\}\right|, \tag{18}$$

where $\text{Top}_K(\hat{y})$ denotes the K sequences ranked highest by the model, y_i is the measured activity of sequence i, and τ_{WT} is the measured wild-type activity for that dataset. This metric directly reports the number of experimentally successful variants among the model's top-K selected designs.

C.2 Enrichment Factor (EF@K)

EF@K is the fraction of active sequences within the model's top-K predictions divided by the fraction of active sequences in the full test pool:

$$\text{EF@}K = \frac{\text{active fraction within top-}K}{\text{active fraction in the full pool}} = \frac{|\text{Top}_K(\hat{y}) \cap \mathcal{A}|/K}{|\mathcal{A}|/n}. \tag{19}$$

This measures how much more likely the model's top-K is to contain an active variant compared to picking K sequences at random from the pool. EF@$K = 1$ means no better than random; EF@$K > 1$ means the model's top-K is enriched for true actives.

C.3 Normalized Discounted Cumulative Gain (NDCG@K)

NDCG additionally rewards the model for placing the *best* sequences *closer to the top* of its ranking, rather than only checking whether they fall somewhere within the top-K set. For datasets with negative activity scores, we shift relevance by the pool's global minimum, $r(x) = y(x) - \min_{x' \in \mathcal{D}_{\text{test}}} y(x')$, which preserves rank order and relative gaps between sequences and does not affect model-to-model comparisons within a dataset; otherwise, $r(x) = y(x)$ directly.

Taking the model's top-K sequences in the order it ranked them, we compute a discounted sum of their relevance, where each position's contribution is divided by a discount that grows with rank (so a good sequence ranked 1st contributes more than the same sequence ranked 10th):

$$\text{DCG@}K = \sum_{j=1}^{K} \frac{r(x_{(j)})}{\log_2(j+1)}, \tag{20}$$

where $x_{(1)}, \ldots, x_{(K)}$ are the model's top-K sequences in the order the model ranked them. We then compute the same quantity for the *ideal* ranking, i.e., the top-K sequences sorted by their true activity, giving the best possible score achievable on this pool:

$$\text{IDCG@}K = \sum_{j=1}^{K} \frac{r(x_{[j]})}{\log_2(j+1)}, \tag{21}$$

where $x_{[1]}, \ldots, x_{[K]}$ are the true top-K sequences sorted by activity, descending. Normalizing gives

$$\text{NDCG@}K = \frac{\text{DCG@}K}{\text{IDCG@}K}, \tag{22}$$

a score in $[0, 1]$ indicating how close the model's own ranking gets to the best possible ranking of the same pool.

C.4 Pearson and Spearman Correlation

We additionally report the Pearson and Spearman correlation between the model's score $\hat{y}$ and the true activity y across the full test pool, measuring how well the model's scores track absolute fitness (Pearson) and relative ranking (Spearman), independent of any specific choice of K.

C.5 Top-K Retrieval

In addition, we report the raw overlap between the model's top-K predictions and the true top-K sequences by activity:

$$\text{Top-}K\ (\%) = \frac{|\text{Top}_K(\hat{y}) \cap \text{Top}_K(y)|}{K} \times 100. \tag{23}$$

Results for this metric are reported in Table 13.

Table 13: Top-K retrieval (%) across four FLIP2 datasets. $K = 10/100$.

Dataset	Method	ProteinMPNN	ESM-IF
Amylase	Base	0 / 19	0 / 23
	+SPPO	0 / 18	0 / 24
	+FPA	0 / 25	10 / 29
	+PIF w/o Neg	0 / 23	10 / 27
	+PIF w/o PFM	10 / 23	10 / 31
	+PIF (Full)	10 / 31	10 / 27
IRED	Base	0 / 7	0 / 7
	+SPPO	0 / 4	0 / 6
	+FPA	0 / 9	0 / 9
	+PIF w/o Neg	0 / 9	0 / 10
	+PIF w/o PFM	0 / 6	0 / 9
	+PIF (Full)	0 / 14	0 / 10
TrpB	Base	0 / 0	0 / 0
	+SPPO	0 / 0	0 / 0
	+FPA	0 / 0	0 / 0
	+PIF w/o Neg	0 / 1	0 / 0
	+PIF w/o PFM	0 / 0	0 / 0
	+PIF (Full)	0 / 4	0 / 0
NucB	Base	0 / 0	0 / 0
	+SPPO	0 / 1	0 / 0
	+FPA	0 / 1	0 / 1
	+PIF w/o Neg	0 / 0	0 / 2
	+PIF w/o PFM	0 / 1	0 / 1
	+PIF (Full)	0 / 3	0 / 2

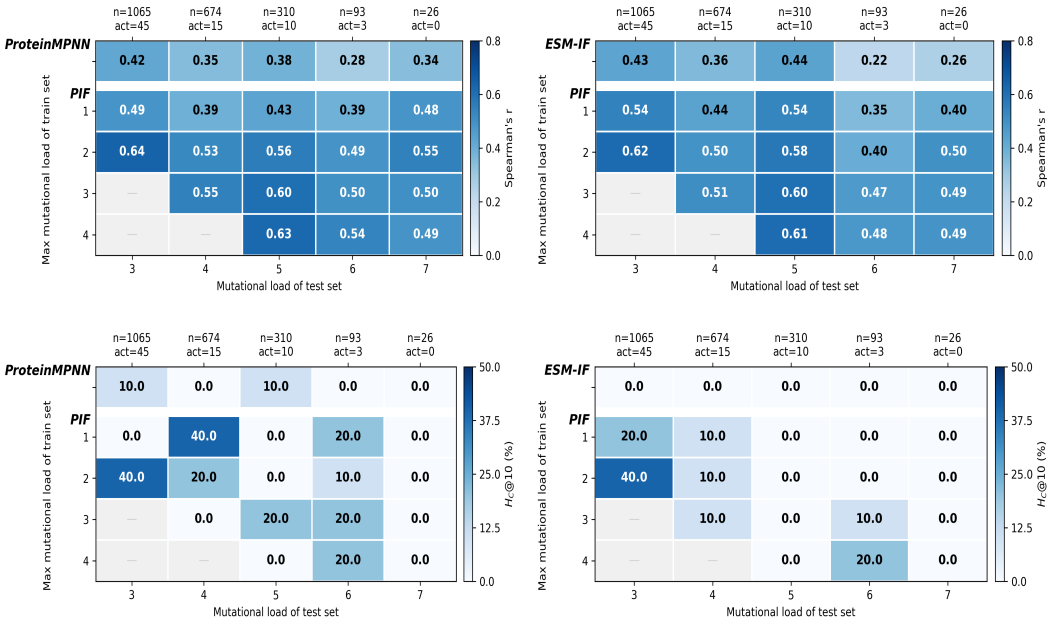


Figure 3: Fine-grained extrapolation on Amylase across training and test mutational loads, measured by Spearman correlation (top) and H_C@10 (bottom), for ProteinMPNN (**Left**) and ESM-IF (**Right**), complementing the Pearson trend in Figure 1. Columns denote test mutational load and rows the maximum mutational load used for PIF fine-tuning. The Baseline row is the vanilla pretrained model; cells with $k_{\text{test}} \leq k_{\text{train}}$ are omitted. Column headers report the number of test variants (n) and active variants exceeding wild-type activity (act).