MCRA: MULTI-CONTEXT RESIDUE ADAPTATION FOR PROTEIN LANGUAGE MODEL TRANSFER

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein language models encode complementary information across layers and sequence contexts, but frozen-backbone transfer often relies on the final layer or a uniform adapter. We introduce MCRA (**M**ulti-**C**ontext **R**esidue **A**daptation), a residue-wise multi-scale framework that learns task-specific layer fusion, routes residues among context experts, and gates the adapted features with the frozen final-layer representation. On 13 protein prediction benchmarks, we compare MCRA with eight frozen protein models equipped with independently trained six-layer Transformer adapters, using the same data splits and evaluation metrics. MCRA obtains the highest or tied-highest score on 11 tasks. Component comparisons examine layer fusion and context adaptation, while perturbation analyses describe task-dependent changes in sequence-averaged routing. Together, these results support coordinating task-level depth selection with residue-level context adaptation for frozen protein model transfer.

1 INTRODUCTION

Protein language models (PLMs) have become general-purpose backbones for protein structure, function, interaction, and engineering tasks by learning transferable representations from sequence data (Rives et al., 2021; Lin et al., 2023; Elnaggar et al., 2022; Chen et al., 2025; Shen et al., 2026; Hayes et al., 2025; Su et al., 2024). As these models continue to grow, effective downstream transfer becomes increasingly important: full-model fine-tuning is costly, whereas frozen-backbone transfer (Schmirler et al., 2024; Vieira et al., 2025; Rao et al., 2019; Sledzieski et al., 2024) offers a practical alternative by adapting pretrained representations with task-specific modules.

PLM transfer must address two complementary forms of heterogeneity. First, protein tasks depend on different ranges of sequence context. Local motifs and physicochemical patterns can be important for residue-level properties, whereas contact prediction and protein–protein interaction prediction require broader residue dependencies; protein-level properties such as solubility and stability may depend on both local and global information (Marks et al., 2011; Ovchinnikov et al., 2014; Hopf et al., 2014; Jones & Thornton, 1996; Hebditch et al., 2017; Khurana et al., 2018). Second, PLM layers encode complementary information, yet many transfer pipelines rely primarily on the final-layer representation or apply the same adaptation operator across sequence positions (Vig et al., 2021; Li et al., 2022). More broadly, evidence from biological sequence modeling indicates that useful context and representation reliability vary across models and downstream tasks (Hirota et al., 2026; Prabakaran & Bromberg, 2026). A uniform transfer strategy therefore cannot explicitly select either task-relevant representation depths or residue-specific context transformations.

Parameter-efficient fine-tuning methods reduce adaptation cost by modifying only a restricted set of parameters (Xu et al., 2026; Houlsby et al., 2019; Hu et al., 2022a; Liu et al., 2024; Zeng et al., 2024). These methods are effective for efficient model adaptation, but they do not typically coordinate task-level selection of pretrained layers with residue-level routing over multiple context operators. Parameter-efficient adaptation and explicit representation selection are therefore complementary objectives. Recent studies further highlight the rich and task-dependent information encoded by protein models, motivating transfer mechanisms that selectively use pretrained representations rather than treating them uniformly (Simon & Zou, 2025; Candido et al., 2026).

We formulate frozen PLM transfer as joint layer-and-scale context selection. MCRA learns task-level weights over pretrained layers, routes residues among context experts with different local con-

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.

volutional profiles, and gates their outputs with the frozen final-layer representation. The methodological contribution lies in coordinating these choices within one protein-transfer architecture: layer selection determines which pretrained features are available, while residue-wise routing determines how they are contextualized. We test the combined design against supervised Transformer adapters on 13 ESM2 tasks, compare adaptation methods with a fixed backbone, and examine transfer across six additional PLMs. Component comparisons and behavioral analyses then investigate how the learned depth and context preferences relate to the task, without requiring updates to pretrained backbone weights. Our main contributions are:

- **Task-dependent depth–context preferences.** We identify distinct preferences for pretrained layers and local context scales across protein prediction tasks, motivating joint adaptation of representation depth and sequence context.
- **Joint depth–context adaptation.** We introduce MCRA, a frozen-backbone transfer framework that integrates task-specific layer fusion, residue-wise multi-scale routing, and gated fusion of pretrained and adapted features.
- **Analysis of learned selection behavior.** We characterize learned depth and context selection through component comparisons, targeted sequence perturbations, and a structural case study, providing insights into task-dependent adaptation behavior.
- **Broad empirical validation.** We evaluate MCRA on 13 protein benchmarks, obtaining the highest or tied-highest point estimates among the evaluated methods on 11 tasks and improving 17 of 18 task–backbone pairs across six additional backbones.

2 Related Work

2.1 Protein Language Model Transfer and Adaptation

PLMs learn transferable biochemical, structural, evolutionary, and functional representations, making downstream adaptation increasingly important as model quality improves (Schmirler et al., 2024). Existing approaches mainly focus on parameter-efficient adaptation or representation enrichment. Adapter (Houlsby et al., 2019), LoRA (Hu et al., 2022a), and DoRA (Liu et al., 2024) update only a restricted set of parameters and have been effective for protein tasks (Zeng et al., 2024), but they do not typically combine task-specific layer selection with residue-wise context routing. Multimodal models such as SaProt (Su et al., 2024), ProTrek (Su et al., 2025), and ESM3 (Hayes et al., 2025) enrich representations with structural or functional information, but require multimodal inputs or supervision.

MCRA is complementary to both directions by selecting task-relevant layers and residue-specific context directly from a frozen PLM, without modifying the backbone or introducing additional modalities.

2.2 Adaptive Multi-Scale Routing

Protein properties depend on information spanning multiple levels, from local sequence and structural motifs to long-range residue contacts and fold-level organization (Mackenzie & Grigoryan, 2017; Joseph & de Brevern, 2014). This hierarchy is also reflected in pretrained representations, where different layers encode complementary biological signals (Hirota et al., 2026). Multi-scale protein models combine features extracted from different context ranges, as exemplified by MMSMAPlus (Wang et al., 2023), whereas mixture-of-experts methods enable conditional computation by routing inputs among specialized experts (Shazeer et al., 2017; Fedus et al., 2022; Riquelme et al., 2021). However, these two directions are rarely integrated to jointly select representation depth and residue-specific context during frozen PLM transfer.

MCRA addresses this gap through task-level layer selection and residue-level routing over multi-scale context experts, where scale denotes the kernel-defined local sequence context applied before global self-attention.

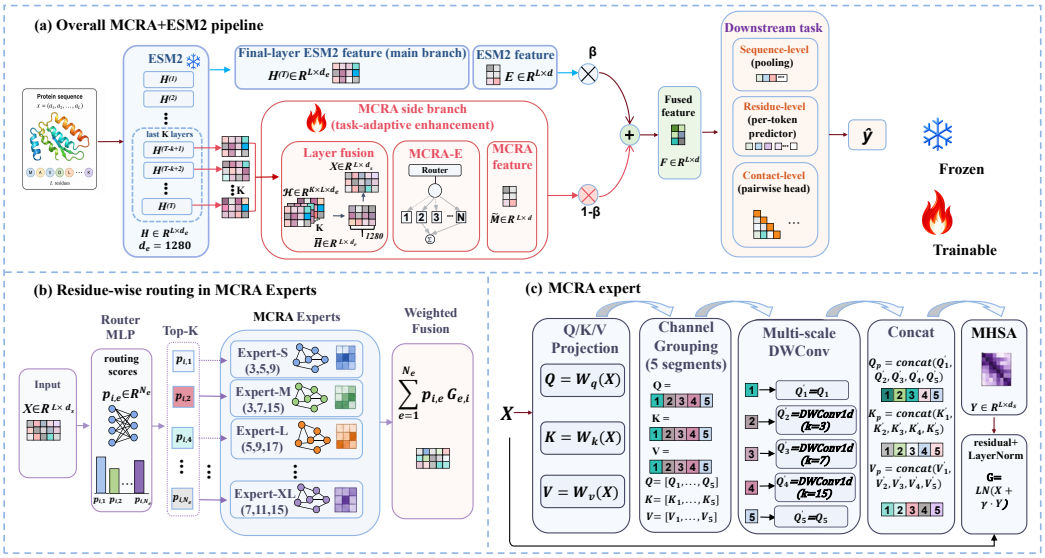


Figure 1: Overview of MCRA for frozen ESM2 transfer with multi-layer fusion, adaptive scale routing, and gated prediction.

3 Methodology

3.1 Problem Formulation and Framework

Given a protein sequence $x = (a_1, \ldots, a_L)$, we predict a task-specific target y defined at the protein, residue, or residue-pair level. An ESM2 encoder produces hierarchical hidden states

$$\mathcal{H}(x) = \{H^{(1)}, \ldots, H^{(T)}\}, \qquad H^{(t)} \in \mathbb{R}^{L \times d_e}, \tag{1}$$

where T and d_e denote the number of Transformer layers and the hidden dimension, respectively. Although ESM2 provides strong general protein representations, downstream accuracy remains sensitive to how these representations are adapted (Lin et al., 2023; Schmirler et al., 2024). We therefore learn a transfer function f_θ,

$$\hat{y} = f_\theta(\mathcal{H}(x)), \tag{2}$$

that explicitly selects both representation hierarchy and biological context scale.

$$\begin{aligned} \{H^{(t)}\}_{t=T-K+1}^{T} &\xrightarrow{\text{layer fusion}} X \xrightarrow{\text{MCRA}} M \\ &\xrightarrow{\text{gated fusion}} F \xrightarrow{\text{task head}} \hat{y}. \end{aligned} \tag{3}$$

where $H^{(t)}$ denotes the t-th ESM2 hidden state, X the layer-fused feature, M the MCRA-adapted feature, F the fused transfer representation, and $\hat{y}$ the prediction. As illustrated in Fig. 1, the design tests whether jointly selecting representation depths and local context profiles improves frozen-backbone transfer.

3.2 Multi-layer Fusion of Task-dependent Representation Hierarchies

Protein language model layers encode complementary biochemical and structural information (Rives et al., 2021; Vig et al., 2021). To integrate this hierarchy rather than relying only on the final layer, we fuse the last K ESM2 states using task-learned normalized weights:

$$\alpha_k = \frac{\exp(s_k)}{\sum_{j=1}^{K} \exp(s_j)}, \qquad \bar{H} = \sum_{k=1}^{K} \alpha_k H^{(T-K+k)}, \tag{4}$$

where $\{s_k\}_{k=1}^{K}$ are trainable scalars. The resulting weights capture task-dependent contributions while retaining complementary information across multiple representation depths. The fused state is projected into the side-branch dimension,

$$X = W_s \bar{H}, \qquad X \in \mathbb{R}^{L \times d_s}, \tag{5}$$

which places the adaptation operators in a separate feature space while keeping the pretrained backbone frozen.

3.3 Multi-Context Residue Adaptation

In MCRA, multi-context adaptation spans representation depth and receptive-field scale: layer fusion selects task-relevant depths, while expert routing selects residue-specific scales. MCRA builds each scale-specific context expert from a modified Group-Mix Attention operator (Ge et al., 2023), assigning expert-specific kernels and adding residue-wise sparse routing across experts. For expert e, each projected attention tensor $U \in \{Q, K, V\}$ is divided along the channel dimension,

$$U = [U_1, \ldots, U_m], \tag{6}$$

and selected groups are transformed by depthwise convolutions with expert-specific kernels:

$$\widetilde{U}_{e,j} = \begin{cases} U_j, & j \in \mathcal{I}, \\ \mathrm{DWConv}_{k_{e,j}}(U_j), & j \in \mathcal{C}. \end{cases} \tag{7}$$

The identity groups preserve unaltered features, whereas the convolutional groups capture residue patterns at different receptive fields. Self-attention then integrates these local responses over the full sequence:

$$G_e = \mathrm{LN}\Big(X + \gamma_e \,\mathrm{Attn}(\widetilde{Q}_e, \widetilde{K}_e, \widetilde{V}_e)\Big). \tag{8}$$

This local–global construction is motivated by the multi-scale organization of protein structure and function (Wang et al., 2023).

The 8 independently parameterized context experts use four repeated scale configurations: Small $(3, 5, 9)$, Medium $(3, 7, 15)$, Large $(5, 9, 17)$, and XLarge $(7, 11, 15)$, with two experts per scale. These labels identify local kernel profiles, not a strict ordering of maximum receptive field or predefined structural categories; every expert also contains global self-attention.

A fixed-scale operator imposes the same context scale on every residue. MCRA instead routes each residue among scale-specific context experts. For residue i, the router computes

$$q_i = \mathrm{softmax}\left(\frac{\mathrm{MLP}(\mathrm{LN}(X_i))}{\tau}\right), \tag{9}$$

where τ controls routing sharpness. When sparse routing is enabled, only the Top-k experts are retained and renormalized:

$$p_{i,e} = \frac{q_{i,e}\,\mathbf{1}[e \in \mathcal{S}_i]}{\sum_{j \in \mathcal{S}_i} q_{i,j}}, \qquad \mathcal{S}_i = \mathrm{TopK}(q_i, k). \tag{10}$$

The adapted residue representation is

$$M_i = \sum_{e=1}^{N_e} p_{i,e} G_{e,i}. \tag{11}$$

This formulation follows sparsely gated expert models (Shazeer et al., 2017; Fedus et al., 2022), but performs residue-wise adaptive scale selection by allocating each residue across experts with different local receptive fields.

3.4 Gated Fusion and Prediction

The final ESM2 state and MCRA output are projected into a shared space:

$$E = W_p H^{(T)}, \qquad \widetilde{M} = W_o M. \tag{12}$$

A residue-wise gate then combines pretrained and scale-adapted information:

$$\beta = \sigma(W_f[E; \widetilde{M}]), \qquad F = \mathrm{LN}\Big(\beta \odot E + (1-\beta) \odot \widetilde{M}\Big). \tag{13}$$

Thus, larger β values favor pretrained features, whereas smaller values favor MCRA features.

For sequence-level tasks, masked mean pooling gives

$$z = \frac{\sum_{i=1}^{L} m_i F_i}{\sum_{i=1}^{L} m_i}, \qquad \hat{y} = h_{\mathrm{seq}}(z), \tag{14}$$

where m_i is the sequence mask. Residue and residue-pair tasks use $\hat{y}_i = h_{\mathrm{res}}(F_i)$ and $\hat{y}_{ij} = h_{\mathrm{pair}}(F_i, F_j)$, respectively.

4 Experiments

4.1 Experimental Setup

We evaluate MCRA on 13 protein prediction benchmarks spanning structure, solubility, localization, interaction, and production-related properties. Within this evaluation, Contact and Fold Cls assess structural prediction at the residue-pair and protein levels, respectively. MCRA is evaluated using the predefined benchmark splits and task-specific metrics. Hyperparameters are selected using the validation set only. Task definitions, evaluation metrics, and split statistics are provided in Appendix A. Every non-MCRA entry in Table 1 uses a frozen PLM followed by a task-specific six-layer Transformer adapter (T-Adapter; hidden size 480, 20 attention heads) and a task output layer. Each model–task pair is trained independently with supervision. T-Adapter in Table 4 is this same adaptation method with ESM2. Training settings are listed in Appendix B.7. The main experiments use ESM2-650M as a backbone. Unless otherwise specified, MCRA uses a maximum sequence length of 1024, the final 16 ESM2 hidden layers, 8 context experts grouped into four receptive-field scales with two experts per scale, a router hidden dimension of 1024, and Top-4 routing. Only the MCRA branch and task-specific prediction head are optimized during downstream training. Further training and implementation details are provided in Appendix B.

4.2 Results and Analysis

We evaluate MCRA from three perspectives: overall transfer performance, the contributions of its architectural components, and the learned layer-and-scale context selection behavior. We first compare MCRA with representative PLMs and adaptation methods, then examine its transfer across backbones and component ablations, and finally analyze routing, layer usage, perturbation stability, and structural context.

(1). MCRA Improves Transfer Across Diverse Protein Tasks. Table 1 compares complete transfer pipelines using common splits and metrics. MCRA achieves the highest test score on 10 tasks and ties on one. Contact shows the largest gain, from 0.7662 to 0.8514 (absolute $+0.0852$). Its leading score on Fold Cls (0.8098) further supports MCRA's effectiveness on structure-related prediction tasks. These results motivate examining how different tasks use representation depth and sequence context. Appendix B.10 discusses task-specific performance patterns.

(2). Transferability across Backbones and Comparison with PEFT. The next comparisons separate changes in the pretrained representation from changes in the adaptation method. Backbone transfer spans residue contacts, solubility, and material production; the ESM2 PEFT comparison spans paired interaction, resistance classification, and sorting-signal prediction. Appendix B.10 discusses their biological scope.

Across six frozen backbones, MCRA improves 17 of 18 task–backbone pairs (Table 2). The exception is ProtST on DeepSoluE, where AUROC decreases from 0.7285 to 0.7194. ProtST learns from both protein sequences and textual descriptions of their properties (Xu et al., 2023). For a whole-protein property such as solubility, this property-aware representation may already be well matched to the task, leaving less room for further context adaptation. Its gains on Contact and Mat. Pro. suggest that the value of MCRA depends not only on the backbone, but also on what the task needs from it.

Table 1: Main results on 13 tasks. All non-MCRA rows use a frozen PLM plus a separately trained six-layer Transformer adapter (T-Adapter). Δ is the relative change vs. the strongest baseline; bold marks the highest test scores.

Model Metric	DeepSol AUROC	Fold Cls Acc.	Contact Top-$L/5$	Human PPI AUROC	Anti. Res. Acc.	Sor. Sig. F1-max	Thermo. AUROC	M. I. Bin. Acc.	DL Bin AUROC	DL2 Multi F1-max	Clo. CLF AUROC	Deep SoluE AUROC	Mat. Pro. Acc.
Sequence-based models													
ESM-2	0.8449	0.7755	0.7176	0.8510	0.6342	0.8703	0.9504	0.7117	0.9062	0.7619	0.8059	**0.7763**	0.8119
VenusPLM	0.8278	0.7546	0.5895	0.8215	0.6460	0.8097	0.9170	0.7020	0.9186	0.7381	0.8317	0.7493	0.8202
ESM-C	0.8417	0.7307	0.7203	0.8396	0.6726	0.8539	0.9495	0.7020	0.9048	0.7540	0.8103	0.7601	0.8101
ProtGPT2	0.7888	0.6472	0.0714	0.7978	0.6844	0.7786	0.9140	0.7117	0.9012	0.7034	0.7773	0.6865	0.7676
PGLM	0.8216	0.7706	0.6345	**0.8769**	0.6726	0.8118	0.9422	**0.7451**	0.9150	0.7477	0.8364	0.7555	0.7950
DPLM	0.8284	0.7945	0.7169	0.8576	0.6873	0.8380	0.9395	0.7006	0.9331	0.7576	0.8125	0.7412	0.8014
Sequence–function-based models													
ProtST	0.8195	0.7258	0.5212	0.8003	0.6342	0.8728	0.9439	0.5153	0.9402	0.7489	0.8071	0.7285	0.6926
Sequence–structure–function-based models													
ESM3	0.7811	0.7252	0.7662	0.7248	0.5841	0.7969	0.8784	0.7033	0.9003	0.6585	0.7739	0.6791	0.7751
MCRA	**0.8481**	**0.8098**	**0.8514**	0.8480	**0.7168**	**0.8830**	**0.9707**	**0.7451**	**0.9425**	**0.7855**	**0.8445**	0.7654	**0.8252**
Δ	↑0.4%	↑1.9%	↑11.1%	↓3.3%	↑4.3%	↑1.2%	↑2.1%	→0.0%	↑0.2%	↑3.1%	↑1.0%	↓1.4%	↑0.6%

Table 2: Transfer across frozen PLMs. Baseline denotes the Transformer-adapter results from Table 1; +MCRA uses the MCRA adaptation pathway.

	Contact (Top-$L/5$)		DeepSoluE (AUROC)		Mat. Pro. (Accuracy)	
Backbone	Baseline	+ MCRA	Baseline	+ MCRA	Baseline	+ MCRA
ESM-C	0.7203	**0.8847**	0.7601	**0.7706**	0.8101	**0.8227**
PGLM	0.6345	**0.8081**	0.7555	**0.7629**	0.7950	**0.8148**
ProtGPT2	0.0714	**0.0749**	0.6865	**0.7580**	0.7676	**0.7859**
DPLM	0.7169	**0.8441**	0.7412	**0.7588**	0.8014	**0.8300**
ProtST	0.5212	**0.6684**	**0.7285**	0.7194	0.6926	**0.7791**
ESM3	0.7662	**0.9065**	0.6791	**0.7690**	0.7751	**0.8277**

With the same ESM2 backbone and similar trainable-parameter budgets, MCRA exceeds the evaluated LoRA and DoRA configurations on these three tasks (Table 3). Appendix B.8 details the settings; the tasks span paired interaction, functional classification, and sorting-signal prediction.

Table 3: PEFT comparison on frozen ESM2.

Task	Metric	LoRA	DoRA	MCRA
Human-PPI	AUROC	0.8210	0.8062	**0.8480**
Anti. Res.	Accuracy	0.6313	0.6195	**0.7168**
Sor. Sig.	F1-max	0.8677	0.8724	**0.8830**

(3). Task-Dependent Local-Context Routing.

The performance comparisons motivate a closer look at how context is allocated. Figure 2 shows that DeepSol, Fold Cls, and Sor. Sig. favor XLarge; Human-PPI, M. I. Bin., and Clo. CLF favor Small; Anti. Res. and Mat. Pro. favor Large; Contact and Thermo. favor Medium. Even the two structural tasks emphasize different profiles, demonstrating task-dependent use of the same expert pool. A preferred profile is the largest component of a mixture, not an exclusive assignment.

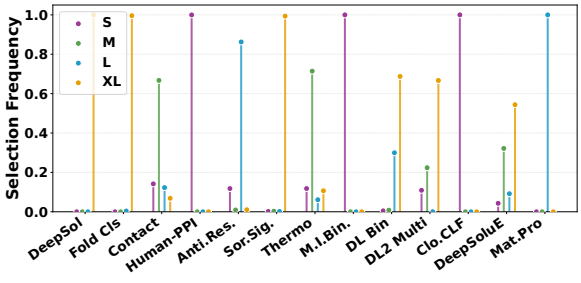


Figure 2: Task-specific routing distributions over four multi-scale context profiles.

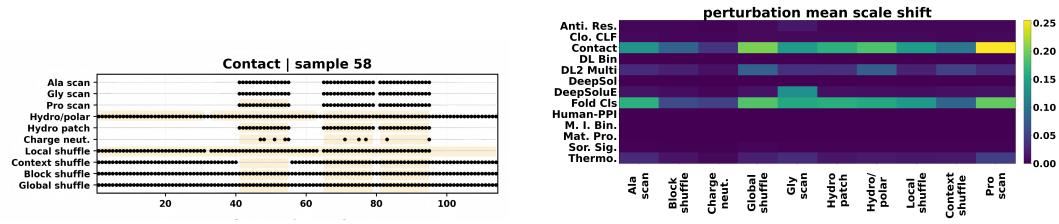


Figure 3: Perturbation design and aggregate routing changes. Left, mutated residues on a representative Contact sequence. Right, mean changes in sequence-averaged scale distributions; darker cells indicate smaller aggregate shifts, not necessarily stable residue-level assignments.

The layer preferences in Fig. 4 complete this picture: tasks select pretrained depths, while residues select context operators. In Table 4, Full improves over task-shared mixing with the same expert pool and layer fusion, supporting coordinated task-level depth selection and residue-wise context adaptation.

(4). Routing Stability under Targeted Sequence Perturbations. Following the patterns above, we examine whether local-context routing remains stable under effective sequence perturbations. We construct paired wild-type (WT)–mutant samples using Ala, Gly, and Pro scans, physicochemical substitutions, charge neutralization, and sequence shuffling. Candidate windows are ranked using an alanine-window probe, and the highest-impact windows are perturbed to provide a non-trivial stress test. A perturbation is considered effective when it lowers the task-specific target score relative to the WT. Figure 3 (left) illustrates the perturbed positions for representative operators on a Contact test sequence, while Fig. 3 (right) summarizes the resulting routing redistribution across tasks. Most tasks show small changes in sequence-averaged routing, with Contact and Fold Cls relatively more sensitive. For Contact, 61.66% of WT–mutant pairs retain their dominant scale, while 38.34% change it; retention varies by starting scale. Mean scale-shift TV is 0.1413 and mean fusion-gate shift is 0.0119; the diagnostic panels are provided in Appendix C.5 (Fig. 9).

(5). Task-Dependent Depth Preferences and Representation Geometry.

After analyzing residue-context scales, we next examine whether downstream tasks also require different depths of the pretrained PLM hierarchy. Fig. 4 shows the learned fusion weights over the last 16 ESM2 layers. Different tasks exhibit distinct layer preferences, indicating that useful transfer signals are not always concentrated in the final ESM2 layer. Structure-related tasks show the clearest deviation: Fold Cls mainly uses earlier-to-middle layers, while Contact places the largest weights on the earliest selected layers. Other tasks show different representation-depth preferences: localization and production-related tasks mainly emphasize middle layers, whereas DeepSol and Human-PPI rely more on later-intermediate layers. DeepSoluE shows a flatter distribution, indicating weaker specialization.

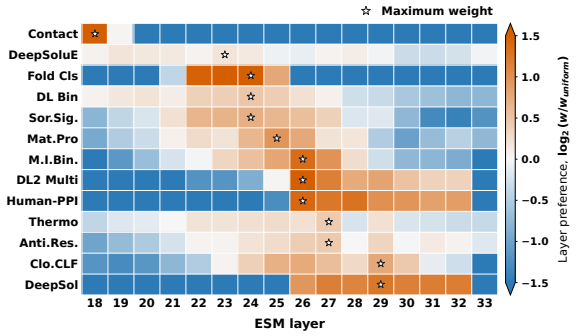


Figure 4: Task-specific layer-fusion preferences across the last 16 ESM2 layers. Colors show $\log_2(w_l/w_{\text{uniform}})$.

Read together, Figs. 2 and 4 reveal task preferences along complementary axes. Contact combines earlier selected layers with a Medium context profile, whereas DeepSol favors later-intermediate layers and XLarge. These choices identify both where useful features are drawn from and how they are processed along the sequence. Replacing residue-wise routing with TF-Mix lowers scores while retaining learned layer fusion, supporting input-dependent contextualization of task-selected features. The representation analysis below illustrates the joint output of these adaptation steps.

Table 4: ESM2 adaptation and component comparisons. T-Adapter is the six-layer Transformer adaptation used by every non-MCRA baseline in Table 1; this row uses ESM2. TF-Mix shares Full's eight-expert pool and Top-4 rule, but learns task-shared weights (Appendix B.11).

Variant Metric	DeepSol AUROC	Fold Cls Acc.	Contact Top-$L/5$	Human PPI AUROC	Anti. Res. Acc.	Sor. Sig. F1-max	Thermo. AUROC	M. I. Bin. Acc.	DL Bin AUROC	DL2 Multi F1-max	Clo. CLF AUROC	Deep SoluE AUROC	Mat. Pro. Acc.
Full	**0.8481**	**0.8098**	**0.8514**	0.8480	**0.7168**	**0.8830**	**0.9707**	**0.7451**	**0.9425**	**0.7855**	**0.8445**	0.7654	0.8252
T-Adapter	0.8449	0.7755	0.7176	**0.8510**	0.6342	0.8703	0.9504	0.7117	0.9062	0.7619	0.8059	**0.7763**	0.8119
TF-Mix	0.8205	0.7982	0.8445	0.8105	0.6873	0.8751	0.9700	0.7437	0.9384	0.7693	0.8253	0.7549	0.8177
w/o Adapt.	0.8052	0.8067	0.8033	0.8317	0.6637	0.8423	0.9702	0.7396	0.9389	0.7674	0.8230	0.7546	**0.8263**
w/o Layers	0.8394	0.7896	0.8401	0.8310	0.6106	0.8494	0.9675	0.7033	0.9313	0.7803	0.8290	0.7580	0.8190
Head only	0.7614	0.5313	0.6652	0.7944	0.5693	0.7057	0.9603	0.5501	0.9167	0.7167	0.7865	0.7547	0.7839

Figure 5 complements the routing and depth analyses by comparing final-layer ESM2 features, the MCRA branch, fused features, and a fixed-scale control. Real proteins and composition-preserving shuffled controls occupy different regions, while the adapted spaces show task-dependent organization. These panels connect the learned context preferences to the resulting task representations. Appendix E.3 details the projection and its interpretation.
For this comparison, residue features are pooled over valid positions using the prediction mask. The joint plots include at most 500 real and 250 shuffled examples per task (seed 42), using cosine-distance t-SNE with perplexity 30 (Appendix E.3). Shuffling preserves sequence length and amino-acid composition, so the control probes sensitivity to residue order rather than composition.

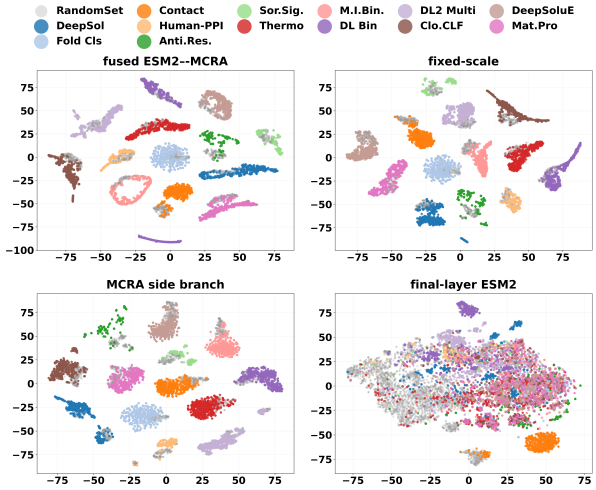


Figure 5: t-SNE visualization of protein representations. Panels show fused ESM2–MCRA, a fixed-scale context adapter, the MCRA side branch, and final-layer ESM2; grey points are shuffled controls.

(6). An Illustrative Structural-Context Case: TEM-1. We further examine whether MCRA routing reflects structural context using TEM-1, a class A serine β-lactamase that hydrolyzes β-lactam antibiotics. Given the TEM-1 sequence (P62593), the Anti. Res.-trained MCRA model correctly predicts the *Betalactams* class with probability 0.999995. Fig. 6 shows Large-scale routing across most of the fold, with localized Small- and Medium-scale patches. Matched DSSP and solvent-accessibility views show that Large routing is strongest at exposed residues, whereas buried residues

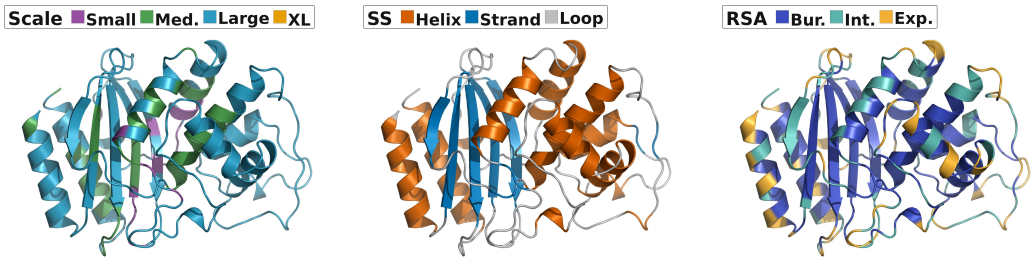


Figure 6: TEM-1: dominant routing scale, DSSP secondary structure, and solvent accessibility (left to right).

Table 5: Parameter statistics and forward-pass computational cost.

Method	Parameter Statistics				Forward-pass Cost		
	Total (M)	Trainable (M)	Active/sample (M)	Active (%)	GFLOPs ↓ per sample	Latency ↓ (ms/sample)	Peak memory ↓ (GiB)
LoRA	775.5	122.1	775.5	100.00	1057.9	44.24 ± 0.87	3.43
DoRA	775.6	123.2	775.6	100.00	1113.4	52.46 ± 1.27	3.45
w/o Adapt.	720.0	67.6	720.0	100.00	–	–	–
MCRA	777.6	125.3	718.6	92.41	**1041.9**	$\mathbf{24.21 \pm 0.72}$	3.92

retain more Medium routing. Thus, TEM-1 provides an illustrative case in which MCRA combines a global broad-context preference with structural-context-dependent scale adjustment. Details are provided in Appendix D.

(7). Task-Dependent Fusion of Frozen and Adapted Features.
Figure 7 compares the fusion dynamics of Fold Cls and Mat. Pro. Fold Cls assigns greater weight to the MCRA branch than to ESM2 and reaches its best validation performance midway through training. In contrast, Mat. Pro. remains dominated by ESM2 and converges earlier. The gate therefore balances frozen and adapted features differently across tasks.
The gate complements depth and scale selection by controlling how much adapted output enters the final representation. Its task-dependent trajectory lets the model retain pretrained information while incorporating contextual features from the side branch (Appendix E.4).

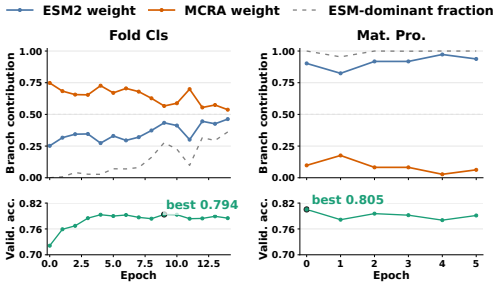


Figure 7: ESM2–MCRA fusion dynamics.

4.3 Ablation Study

Table 4 examines adaptation, representation depth, and expert allocation using the Table 1 training protocols. Full exceeds T-Adapter on 11 tasks and the fixed-scale operator on 12; final-layer-only input also reduces scores. Task-Fixed-Multi (TF-Mix) retains the eight multi-scale experts, learned layer fusion, and final gate, but shares one learned mixture across all residues and proteins within a task. Both Full and TF-Mix select Top-4 experts under the same training settings. Full obtains higher scores on all 13 tasks, supporting input-dependent allocation beyond a task-wide scale preference. Together with the depth analysis, this supports coordinating task-level representation selection with residue-wise contextualization. Appendix B.11 specifies the comparison.

4.4 Capacity and Forward-Compute Trade-off

Following these ablations, we compare MCRA, LoRA, and DoRA under closely matched trainable-parameter budgets (Table 5). MCRA retains its gains on the three evaluated tasks (Table 3).

Within this comparison, MCRA uses 1.51% fewer GFLOPs and is $1.83\times$ faster than LoRA, with 14.29% higher peak memory, under the forward-profiling workload specified in Appendix B.9.

5 Conclusion

MCRA coordinates task-level depth selection with residue-wise multi-scale routing for frozen PLM transfer. Results across 13 benchmarks, component comparisons, and learned preferences support the value of adapting both representation depth and sequence context. Future work will extend transfer to more tasks and backbones, improve predictive performance, develop lighter expert modules with faster execution, and evaluate longer proteins beyond the current input budget.

REFERENCES

José Juan Almagro Armenteros, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. Deeploc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, 33(21):3387–3395, 11 2017. ISSN 1367-4803. doi: 10.1093/bioinformatics/btx431. URL https://doi.org/10.1093/bioinformatics/btx431.

Salvatore Candido, Thomas Hayes, Alexander Derry, Roshan Rao, Zeming Lin, Robert Verkuil, Bryan Z. Wu, Jin Sub Lee, Elise S. Bruguera, Jehan A. Keval, Mykhailo Kopylov, John E. Pak, Wesley Wu, Neil Thomas, Samson Mataraso, Alvin Hsu, Ashton C. Trotman-Grant, Kilian Fatras, Allan dos Santos Costa, Rohil Badkundri, Halil Akın, Deniz Oktay, Jonathan Deaton, Elizabeth Montabana, Hrishita Sitwala, Yue Yu, Marius Wiggert, Dylan Alexander Carlin, Anthony W. Goering, Tomasz Blazejewski, McCullen Sandora, Michael Hla, Tina Z. Jia, Leon H. Kloker, Nicholas J. Sofroniew, Masatoshi Uehara, Jassi Pannu, Sharrol Bachas, Daniel S. Liu, Tom Sercu, and Alexander Rives. Language modeling materializes a world model of protein biology. *bioRxiv*, 2026. doi: 10.64898/2026.06.03.729735. URL https://www.biorxiv.org/content/early/2026/06/04/2026.06.03.729735.

Bo Chen, Xingyi Cheng, Pan Li, Yangli-ao Geng, Jing Gong, Shen Li, Zhilei Bei, Xu Tan, Boyan Wang, Xin Zeng, Chiming Liu, Aohan Zeng, Yuxiao Dong, Jie Tang, and Le Song. xTrimoPGLM: unified 100-billion-parameter pretrained transformer for deciphering the language of proteins. *Nature Methods*, 22(5):1028–1039, May 2025. ISSN 1548-7105. doi: 10.1038/s41592-025-02636-z. URL https://doi.org/10.1038/s41592-025-02636-z.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL http://jmlr.org/papers/v23/21-0998.html.

Chongjian Ge, Xiaohan Ding, Zhan Tong, Li Yuan, Jiangliu Wang, Yibing Song, and Ping Luo. Advancing vision transformers with group-mix attention, 2023. URL https://arxiv.org/abs/2311.15157.

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J. Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf A. Khan, Chetan Mishra, Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025. doi: 10.1126/science.ads0018. URL https://www.science.org/doi/abs/10.1126/science.ads0018.

Max Hebditch, M. Alejandro Carballo-Amador, Spyros Charonis, Robin Curtis, and Jim Warwicker. Protein-Sol: a web tool for predicting protein solubility from sequence. *Bioinformatics (Oxford, England)*, 33(19):3098–3100, October 2017. ISSN 1367-4811 1367-4803. doi: 10.1093/bioinformatics/btx345.

Keisuke Hirota, Koichi Higashi, Ken Kurokawa, and Takuji Yamada. Benchmarking long-context genome language models on biosynthetic gene clusters. *bioRxiv*, 2026. doi: 10.64898/2026.05.12.724296. URL https://www.biorxiv.org/content/early/2026/05/15/2026.05.12.724296.

Thomas A Hopf, Charlotta P I Schärfe, João P G L M Rodrigues, Anna G Green, Oliver Kohlbacher, Chris Sander, Alexandre M J J Bonvin, and Debora S Marks. Sequence co-evolution gives 3d contacts and structures of protein complexes. *eLife*, 3:e03430, sep 2014. ISSN 2050-084X. doi: 10.7554/eLife.03430. URL https://doi.org/10.7554/eLife.03430.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022a. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Mingyang Hu, Fajie Yuan, Kevin K Yang, Fusong Ju, Jin Su, Hui Wang, Fei Yang, and Qiuyang Ding. Exploring evolution-aware &-free protein language models as protein function predictors. *arXiv preprint arXiv:2206.06583*, 2022b.

Tim J. P. Hubbard, Bart Ailey, Steven E. Brenner, Alexey G. Murzin, and Cyrus Chothia. Scop: a structural classification of proteins database. *Nucleic Acids Research*, 27(1):254–256, 01 1999. ISSN 0305-1048. doi: 10.1093/nar/27.1.254. URL https://doi.org/10.1093/nar/27.1.254.

Anna Jarzab, Nils Kurzawa, Thomas Hopf, Matthias Moerch, Jana Zecha, Niels Leijten, Yangyang Bian, Eva Musiol, Melanie Maschberger, Gabriele Stoehr, Isabelle Becher, Charlotte Daly, Patroklos Samaras, Julia Mergner, Britta Spanier, Angel Angelov, Thilo Werner, Marcus Bantscheff, Mathias Wilhelm, Martin Klingenspor, Simone Lemeer, Wolfgang Liebl, Hannes Hahne, Mikhail M. Savitski, and Bernhard Kuster. Meltome atlas—thermal proteome stability across the tree of life. *Nature Methods*, 17(5):495–503, May 2020. ISSN 1548-7105. doi: 10.1038/s41592-020-0801-4. URL https://doi.org/10.1038/s41592-020-0801-4.

S. Jones and J. M. Thornton. Principles of protein-protein interactions. *Proceedings of the National Academy of Sciences of the United States of America*, 93(1):13–20, January 1996. ISSN 0027-8424 1091-6490. doi: 10.1073/pnas.93.1.13.

Agnel Praveen Joseph and Alexandre G. de Brevern. From local structure to a global framework: recognition of protein folds. *Journal of The Royal Society Interface*, 11, 2014. URL https://api.semanticscholar.org/CorpusID:2596485.

Sameer Khurana, Reda Rawi, Khalid Kunji, Gwo-Yu Chuang, Halima Bensmail, and Raghvendra Mall. Deepsol: a deep learning framework for sequence-based protein solubility prediction. *Bioinformatics*, 34(15):2605–2613, 08 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty166. URL https://doi.org/10.1093/bioinformatics/bty166.

Francesca-Zhoufan Li, Ava P Amini, Kevin K Yang, and Alex X Lu. Pretrained protein language model transfer learning: is the final layer representation what we want. *Proc. Mach. Learn. for Struct. Biol. Work. NeurIPS*, 2022, 2022.

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. URL https://www.science.org/doi/abs/10.1126/science.ade2574.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pp. 32100–32121. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/liu24bn.html.

Craig O Mackenzie and Gevorg Grigoryan. Protein structural motifs in prediction and design. *Current Opinion in Structural Biology*, 44:161–167, 2017. ISSN 0959-440X. doi: https://doi.org/10.1016/j.sbi.2017.03.012. URL https://www.sciencedirect.com/science/article/pii/S0959440X16301865.

Debora S. Marks, Lucy J. Colwell, Robert Sheridan, Thomas A. Hopf, Andrea Pagnani, Riccardo Zecchina, and Chris Sander. Protein 3D structure computed from evolutionary sequence variation. *PloS one*, 6(12):e28766, 2011. ISSN 1932-6203. doi: 10.1371/journal.pone.0028766.

Sergey Ovchinnikov, Hetunandan Kamisetty, and David Baker. Robust and accurate prediction of residue–residue interactions across protein interfaces using evolutionary information. *eLife*, 3: e02030, may 2014. ISSN 2050-084X. doi: 10.7554/eLife.02030. URL https://doi.org/10.7554/eLife.02030.

Xiao-Yong Pan, Ya-Nan Zhang, and Hong-Bin Shen. Large-scale prediction of human protein-protein interactions from amino acid sequence based on latent topic features. *Journal of proteome research*, 9(10):4992—5001, October 2010. ISSN 1535-3893. doi: 10.1021/pr100618t. URL https://doi.org/10.1021/pr100618t.

R. Prabakaran and Yana Bromberg. Quantifying uncertainty in protein representations across models and tasks. *Nature Methods*, 23(4):796–804, April 2026. ISSN 1548-7105. doi: 10.1038/s41592-026-03028-7. URL https://doi.org/10.1038/s41592-026-03028-7.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/37f65c068b7723cd7809ee2d31d7861c-Paper.pdf.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8583–8595. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/48237d9f2dea8c74c2a72126cf63d933-Paper.pdf.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021. doi: 10.1073/pnas.2016239118. URL https://www.pnas.org/doi/abs/10.1073/pnas.2016239118.

Robert Schmirler, Michael Heinzinger, and Burkhard Rost. Fine-tuning protein language models boosts predictions across diverse tasks. *Nature Communications*, 15(1):7407, August 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-51844-2. URL https://doi.org/10.1038/s41467-024-51844-2.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017. URL https://arxiv.org/abs/1701.06538.

Lingdong Shen, Linlin Chao, Tang Liu, Qi Liu, Guowei Zhou, Hao Wang, Xinyue Dong, Tianhong Li, Xiaoming Zhang, and Jinren Ni. Proteinsage: From implicit learning to explicit structural constraints for efficient protein language modeling. *bioRxiv*, 2026. doi: 10.64898/2026.03.17.712034. URL https://www.biorxiv.org/content/early/2026/03/19/2026.03.17.712034.

Elana Simon and James Zou. InterPLM: discovering interpretable features in protein language models via sparse autoencoders. *Nature Methods*, 22(10):2107–2117, October 2025. ISSN 1548-7105. doi: 10.1038/s41592-025-02836-7. URL https://doi.org/10.1038/s41592-025-02836-7.

Samuel Sledzieski, Meghana Kshirsagar, Minkyung Baek, Rahul Dodhia, Juan Lavista Ferres, and Bonnie Berger. Democratizing protein language models with parameter-efficient fine-tuning. *Proceedings of the National Academy of Sciences*, 121(26):e2405840121, 2024. doi: 10.1073/pnas.2405840121. URL https://www.pnas.org/doi/abs/10.1073/pnas.2405840121.

Jin Su, Chenchen Han, Yuyang Zhou, Junjie Shan, Xibin Zhou, and Fajie Yuan. Saprot: Protein language modeling with structure-aware vocabulary. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=6MRm3G4NiU.

Jin Su, Yan He, Shiyang You, Shiyu Jiang, Xibin Zhou, Xuting Zhang, Yuxuan Wang, Xining Su, Igor Tolstoy, Xing Chang, Hongyuan Lu, and Fajie Yuan. A trimodal protein language model enables advanced protein searches. *Nature Biotechnology*, October 2025. ISSN 1546-1696. doi: 10.1038/s41587-025-02836-0. URL https://doi.org/10.1038/s41587-025-02836-0.

Vineet Thumuluri, José Juan AlmagroǍrmenteros, AlexanderǍRosenberg Johansen, Henrik Nielsen, and Ole Winther. Deeploc 2.0: multi-label subcellular localization prediction using protein language models. *Nucleic Acids Research*, 50(W1):W228–W234, 07 2022. ISSN 0305-1048. doi: 10.1093/nar/gkac278. URL https://doi.org/10.1093/nar/gkac278.

Luiz C. Vieira, Morgan L. Handojo, and Claus O. Wilke. Medium-sized protein language models perform well at transfer learning on realistic datasets. *Scientific Reports*, 15(1):21400, July 2025. ISSN 2045-2322. doi: 10.1038/s41598-025-05674-x. URL https://doi.org/10.1038/s41598-025-05674-x.

Jesse Vig, Ali Madani, Lav R. Varshney, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. Bertology meets biology: Interpreting attention in protein language models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=YWtLZvLmud7.

Chao Wang and Quan Zou. Prediction of protein solubility based on sequence physicochemical patterns and distributed representation information with deepsolue. *BMC Biology*, 21:12, 2023. doi: 10.1186/s12915-023-01510-8. URL https://doi.org/10.1186/s12915-023-01510-8.

Huilin Wang, Mingjun Wang, Hao Tan, Yuan Li, Ziding Zhang, and Jiangning Song. Predpcrys: Accurate prediction of sequence cloning, protein production, purification and crystallization propensity from protein sequences using multi-step heterogeneous feature fusion and selection. *PLoS ONE*, 9(8):e105902, 2014. doi: 10.1371/journal.pone.0105902. URL https://doi.org/10.1371/journal.pone.0105902.

Zhongyu Wang, Zhaohong Deng, Wei Zhang, Qiongdan Lou, Kup-Sze Choi, Zhisheng Wei, Lei Wang, and Jing Wu. Mmsmaplus: a multi-view multi-scale multi-attention embedding model for protein function prediction. *Briefings in Bioinformatics*, 24(4):bbad201, 07 2023. ISSN 1477-4054. doi: 10.1093/bib/bbad201. URL https://doi.org/10.1093/bib/bbad201.

Lingling Xu, Haoran Xie, S. Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(6):6107–6126, 2026. doi: 10.1109/TPAMI.2026.3657354.

Minghao Xu, Xinyu Yuan, Santiago Miret, and Jian Tang. ProtST: Multi-modality learning of protein sequences and biomedical texts. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38749–38767. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/xu23t.html.

Jianyi Yang, Ivan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov, and David Baker. Improved protein structure prediction using predicted interresidue orientations. *Proceedings of the National Academy of Sciences*, 117(3):1496–1503, 2020. doi: 10.1073/pnas.1914677117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1914677117.

Shuai Zeng, Duolin Wang, Lei Jiang, and Dong Xu. Parameter-efficient fine-tuning on large protein language models improves signal peptide prediction. *Genome Research*, 34(9):1445–1454, 2024. doi: 10.1101/gr.279132.124. URL http://genome.cshlp.org/content/genome/34/9/1445.

SUPPLEMENTARY APPENDIX

This appendix supplements *MCRA: Multi-Context Residue Adaptation for Protein Language Model Transfer*.

Appendix A describes the transfer tasks, predefined data partitions, input processing, and evaluation metrics. Appendix B specifies the executed model configuration, optimization procedure, checkpoint selection, randomization controls, and compute environment. Appendix C provides a comprehensive description of the perturbation-analysis framework, including the sampling protocol, perturbation operators, saliency-guided window selection, task-aware scoring, routing stability metrics, and task-level diagnostic analyses used to evaluate MCRA's prediction sensitivity and adaptive scale-routing behavior. Appendix D describes the TEM-1 post hoc structural analysis, and Appendix E documents the task-level interpretability and stability analyses of adaptive scale and depth selection, including expert-group and scale preferences, layer-fusion preferences, representation-space shuffled controls, and fusion-gate dynamics.

A DETAILED TRANSFER TASKS AND DATA SPLITS

A.1 TASK SCOPE AND SAMPLE UNITS

The 13 transfer tasks follow the benchmark definitions summarized in Table 6. A sample denotes one protein for all sequence-level tasks, one protein pair for Human-PPI, and one protein together with a residue-pair label matrix for Contact. Fold Cls and Anti. Res. are multiclass tasks; Sor. Sig. and DL2 Multi are multilabel tasks; the remaining sequence-level tasks are binary classification tasks. MCRA receives amino-acid sequences only. Coordinates, contact-map features, DSSP labels, solvent accessibility, and other structural annotations are not used as model inputs in the 13-task transfer experiments.

A.2 PREDEFINED PARTITIONS AND TEST-SET ISOLATION

Each task is distributed as separate training, validation, and test files. We used these predefined partitions without randomly repartitioning examples or moving samples between splits. The training loader shuffles only the order of training examples; validation and test loaders preserve a deterministic order. Hyperparameters, early stopping, and checkpoint selection use validation data only. The held-out test partition is evaluated after training with the selected validation checkpoint and is not used to update model parameters or select a checkpoint. Table 7 reports the number of rows loaded from the files used in the experiments. We did not apply additional oversampling, class rebalancing, or sequence augmentation during transfer training.

The experiment partitions impose a protein sequence-similarity threshold below 30% between training, validation, and test splits. This is a cross-split constraint, not a claim that every pair within a split has similarity below 30%. We keep the resulting split files fixed for all methods. Dataset citations identify the task sources; they do not imply that our split files reproduce every original publication's partitioning scheme. Sequence-similarity control alone does not establish a family- or superfamily-disjoint evaluation.

Table 6: Overview of the selected transfer tasks. All metrics are higher-is-better.

Task	Category	Metric	Task	Category	Metric
DeepSol	Solubility	AUROC	Fold Cls	Structure	Accuracy
Contact	Structure	Top-$L/5$	Anti. Res.	Production	Accuracy
Human-PPI	Interaction	AUROC	Thermo.	Production	AUROC
Sor. Sig.	Localization	F1-max	M. I. Bin.	Interaction	Accuracy
DL Bin	Localization	AUROC	DL2 Multi	Localization	F1-max
Clo. CLF	Production	AUROC	DeepSoluE	Solubility	AUROC
Mat. Pro.	Production	Accuracy			

Table 7: Dataset splits for the 13 transfer tasks used in the experiments. Train, validation, and test columns report the number of samples in each split.

Task	Category	Metric	Train	Val	Test	Total
DeepSol (Khurana et al., 2018)	Solubility	AUROC	55465	6932	6934	69331
DeepSoluE (Wang & Zou, 2023)	Solubility	AUROC	11627	1452	1454	14533
Fold Cls (Hubbard et al., 1999)	Structure	Accuracy	13034	1628	1630	16292
Contact (Yang et al., 2020)	Structure	Top-$L/5$	12005	1500	1501	15006
Human-PPI (Pan et al., 2010)	Interaction	AUROC	30133	270	195	30598
M. I. Bin. (Hu et al., 2022b)	Interaction	Accuracy	5740	717	718	7175
Sor. Sig. (Thumuluri et al., 2022)	Localization	F1-max	1484	185	186	1855
DL Bin (Almagro Armenteros et al., 2017)	Localization	AUROC	6887	846	848	8581
DL2 Multi (Thumuluri et al., 2022)	Localization	F1-max	21949	2743	2744	27436
Anti. Res. (Hu et al., 2022b)	Production	Accuracy	2703	336	339	3378
Thermo. (Jarzab et al., 2020)	Production	AUROC	33474	4184	4184	41842
Clo. CLF (Wang et al., 2014)	Production	AUROC	22223	2777	2778	27778
Mat. Pro. (Wang et al., 2014)	Production	Accuracy	22196	2773	2775	27744

A.3 Sequence processing

The configured model-input budget is 1024 token positions. The tokenizer adds its standard sequence boundary tokens; inputs exceeding the budget are deterministically truncated from the sequence end, while shorter inputs are dynamically padded within a batch. Attention masks exclude padding positions from representation pooling and from token-level routing summaries. For sequence-level tasks, the prediction head receives a masked mean over valid residue representations. Human-PPI is processed by the paired-sequence task interface, and Contact retains residue-wise representations for residue-pair prediction. No structure-derived or physicochemical auxiliary features are enabled in the reported 13-task MCRA configuration.

A.4 Evaluation metrics

All reported metrics are higher-is-better and are computed on the complete held-out test partition. Accuracy is the fraction of correctly predicted examples, using an argmax decision for multiclass tasks and a sigmoid threshold of 0.5 for binary tasks. AUROC is computed from the continuous positive-class probabilities rather than thresholded predictions.

For multilabel tasks, F1-max is the maximum F1 score over decision thresholds,

$$\mathrm{F1}_{\max} = \max_{\tau} \frac{2P(\tau)R(\tau)}{P(\tau) + R(\tau)}, \tag{15}$$

where τ is a score threshold and $P(\tau)$ and $R(\tau)$ are the resulting precision and recall. This follows the task definition and avoids fixing one threshold for tasks with different label prevalences.

For Contact, let L denote the valid protein length. Candidate residue pairs are restricted to valid, upper-triangular positions with sequence separation $|i-j| \geq 6$. Pairs are ranked by predicted contact probability, and Top-$L/5$ is the precision among the highest-scoring $K = \min\{\lfloor L/5 \rfloor, N_{\text{eligible}}\}$ pairs, where N_{eligible} is the number of valid candidate pairs. The reported value is averaged over proteins. Padding positions and short-range pairs excluded by the separation rule do not contribute.

B Training and Implementation Details

Appendices B.1–B.6 describe MCRA and its component controls. The six-layer Transformer baselines use the distinct settings in Appendix B.7; LoRA and DoRA are specified in Appendix B.8. Appendix B.11 documents the routing-granularity comparison under the main-table protocol. Shared data and metrics should not be interpreted as identical adaptation architectures or optimization settings.

Table 8: Common MCRA configuration used for the main 13-task transfer experiments.

Component	Executed setting
Backbone	ESM2-650M, frozen during downstream training
Layer fusion	Last 16 ESM2 layers; learned softmax-normalized scalar weights, initialized uniformly
Side representation	Hidden dimension 1920; 12 attention heads; dropout 0.1
MCRA context experts	8 independently parameterized experts, two experts per scale group
Scale kernels	Small $(3, 5, 9)$; Medium $(3, 7, 15)$; Large $(5, 9, 17)$; XLarge $(7, 11, 15)$
Router	Learned residue-wise MLP; hidden dimension 1024; temperature 1.0; Top-4 experts
Fusion	Residue-wise scalar gate between frozen ESM2 and scale-adapted MCRA features
Routing regularization	Entropy weight 1.0; expert-balance weight 10^{-2}
Input budget	1024 token positions
Auxiliary inputs	No structure, contact-map, or physicochemical feature input

B.1 Backbone and trainable components

The backbone is ESM2-650M (33 Transformer layers; 1280-dimensional hidden states). MCRA trains its adaptation modules and task predictor, while the pretrained ESM2 parameters remain frozen.

The four expert scales are repeated in the implementation in the order Medium, Large, Small, and XLarge, giving two experts for each scale. The released expert implementation divides each projected attention tensor into $m = 5$ channel groups: the first and fifth use identity transformations, and the middle three use the listed depthwise-convolution kernels. Each expert has a trainable scalar residual coefficient γ_e, initialized to zero. The router temperature is fixed at $\tau = 1.0$ in this configuration.

B.2 Task heads, losses, and batch sizes

Sequence-level tasks use masked mean pooling followed by a task-specific linear prediction head. Human-PPI uses the paired-input task head. Contact uses a residue-pair head and a masked pairwise binary objective. Losses are unweighted: we do not introduce task-specific class weights or resampling. The executed per-GPU batch size is 2 for Contact and 8 for every other task, including Thermo.; no gradient accumulation is used. Four data-loader workers are used per training job.

B.3 Optimization and checkpoint selection

For MCRA and its component controls, trainable parameters are optimized with AdamW using learning rate 1×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.98)$, $\epsilon = 10^{-8}$, and weight decay 0. The learning rate follows cosine annealing over the full number of optimization steps without a warm-up stage. Gradients are clipped to a global norm of 1.0. Training runs for at most 50 epochs, with validation after every epoch. Early stopping uses patience 5 and maximizes the task-specific validation metric in Table 9. The three best validation checkpoints and the final checkpoint are retained. After training, the checkpoint with the best monitored validation metric is loaded and evaluated on the held-out test partition.

B.4 Training Objective

Training minimizes the task loss together with router entropy and load-balancing regularization. Let $p_{i,e}$ denote the routing probability assigned to expert e at valid residue position i, let L be the number of valid non-padding positions in the sequence, and let N_e be the number of experts. The entropy

Table 9: MCRA task outputs, losses, per-GPU batch sizes, and checkpoint-selection metrics. Transformer-adapter baselines instead use batch size 64 (Appendix B.7).

Task	Output type	Output dim.	Training loss	Batch	Validation selection metric
DeepSol	Binary	1	BCE with logits	8	AUROC
DeepSoluE	Binary	1	BCE with logits	8	AUROC
Fold Cls	Multiclass	1195	Cross-entropy	8	Accuracy
Contact	Residue pair	1	Masked pairwise BCE	2	Top-$L/5$
Human-PPI	Pair binary	1	BCE with logits	8	AUROC
M. I. Bin.	Binary	1	BCE with logits	8	Accuracy
Sor. Sig.	Multilabel	9	BCE with logits	8	F1-max
DL Bin	Binary	1	BCE with logits	8	AUROC
DL2 Multi	Multilabel	10	BCE with logits	8	F1-max
Anti. Res.	Multiclass	19	Cross-entropy	8	Accuracy
Thermo.	Binary	1	BCE with logits	8	AUROC
Clo. CLF	Binary	1	BCE with logits	8	AUROC
Mat. Pro.	Binary	1	BCE with logits	8	Accuracy

term and the mean expert-usage penalty are

$$\begin{aligned} \mathcal{L}_{\text{ent}} &= -\frac{1}{L}\sum_{i=1}^{L}\sum_{e=1}^{N_e} p_{i,e}\log p_{i,e}, \\ \bar{p}_e &= \frac{1}{L}\sum_{i=1}^{L} p_{i,e}, \qquad \mathcal{L}_{\text{bal}} = \sum_{e=1}^{N_e}\left(\bar{p}_e - \frac{1}{N_e}\right)^2. \end{aligned} \tag{16}$$

The total objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{ent}}\mathcal{L}_{\text{ent}} + \lambda_{\text{bal}}\mathcal{L}_{\text{bal}}. \tag{17}$$

Here $\mathcal{L}_{\text{task}}$ is the task-specific loss in Table 9. We use $\lambda_{\text{ent}} = 1.0$ and $\lambda_{\text{bal}} = 10^{-2}$ in the main 13-task experiments, matching the routing-regularization settings in Table 8. Minimizing $\mathcal{L}_{\text{ent}}$ encourages lower-entropy, sharper routing distributions, whereas $\mathcal{L}_{\text{bal}}$ discourages collapse onto a small subset of experts by penalizing deviations from uniform mean expert usage. The balancing term is computed over valid residue positions so padding tokens do not contribute to the estimated expert load.

B.5 Randomness and result reporting

The main 13-task MCRA runs and their component comparisons use seed 2024. Full, TF-Mix, and the other component controls follow the same predefined splits, optimization schedule, validation-based checkpoint rule, and evaluation implementation. T-Adapter reuses the corresponding ESM2 Transformer-adapter protocol from the main table (Appendix B.7). Each reported task score is the held-out test metric of the checkpoint selected by validation performance. Bold entries identify the highest scores within each comparison. Timing deviations in the efficiency table summarize repeated profiling rounds.

B.6 Hardware and software

All experiments were conducted on a single compute node equipped with an AMD Ryzen Threadripper PRO 7975WX 32-core CPU, 251 GiB of system memory, and two NVIDIA A100-SXM4 GPUs with 80 GiB of memory each. Each main MCRA training job and its component controls, including Full and TF-Mix, used both A100 GPUs with the same task-specific per-GPU batch sizes. The node ran Ubuntu 24.04.3 LTS with Linux kernel 6.17.0-35-generic and NVIDIA driver 580.95.05.

Training used bfloat16 precision and DeepSpeed ZeRO Stage 2. The software environment consisted of Python 3.11.9, CUDA 11.8.89, PyTorch Lightning 2.5.0.post0, Transformers 4.48.1, DeepSpeed 0.16.3, PEFT 0.14.0, TorchMetrics 1.6.1, Torch Geometric 2.6.1, NumPy 1.26.4, pandas 2.2.3, scikit-learn 1.6.1, Hydra 1.3.2, and Matplotlib 3.10.1.

Table 10: Supervised Transformer-adapter settings for every non-MCRA model in the main-results table. Each backbone–task pair is trained independently.

Setting	Value
Pretrained feature extractor	Frozen
Adapter	6-layer Transformer
Hidden size / attention heads	480 / 20
Trainable components	Transformer adapter and task-specific output modules
Optimizer	AdamW
Maximum training epochs	50
Training batch size	64
Learning-rate search	$\{10^{-5}, 10^{-4}\}$, selected on validation data
Early stopping	Patience of 5 epochs
Test evaluation	Held-out test split, with validation-selected configuration

B.7 Baseline and ablation comparison protocol

The comparisons address two different questions: whether MCRA improves a supervised frozen-PLM transfer pipeline, and which parts of its adaptation pathway contribute to the observed scores. All methods use common task partitions and evaluation metrics. Their trainable modules and optimization settings must nevertheless be distinguished.

Transformer-adapter baselines. Every non-MCRA row in Table 1, including ESM2, combines a frozen pretrained feature extractor with a six-layer Transformer adapter and a task-specific output module. A separate supervised model is trained for each backbone–task pair; parameters are not shared across tasks. The adapter has hidden size 480 and 20 attention heads. Its attention, feed-forward, and layer-normalization parameters, together with the task output parameters, are optimized while the PLM remains frozen. Table 10 gives the common baseline settings. Here, "T-Adapter" abbreviates Transformer adapter tuning: an external stack operating on pretrained features, not bottleneck adapters inserted into backbone layers. It is also distinct from linear probing, which replaces that stack with a linear predictor. No non-MCRA entry in the main-results table is a linear probe or a zero-shot result.

Head-only supervision and the baseline discrepancy. Head only, previously labelled w/o MCRA, trains only the task predictor on frozen final-layer ESM2 features. It is supervised, not zero-shot. This path bypasses the Transformer adapter as well as MCRA's layer fusion, context experts, projections, normalization, and fusion gate. The predictor receives 1280-dimensional ESM2 features; the full MCRA model instead produces 480-dimensional fused features through a 1920-dimensional side branch. Sequence-level tasks use a linear predictor after pooling, while Human-PPI and Contact retain their task-specific prediction interfaces. Thus, Head only is a reference for removing the complete adaptation pathway, not an isolated test of adaptive routing.

This architectural difference explains why the ESM2 baseline and Head only are not numerically interchangeable. For example, their Fold Cls scores are 0.7755 and 0.5313, respectively. To make the comparison explicit, T-Adapter in Table 4 reproduces all 13 ESM2 scores from Table 1; it is not a new training run. Against these two references, the full model's Fold Cls gain is 0.0343 and 0.2785, respectively. The former compares adaptation methods; the latter measures the benefit of the whole adaptation pathway over head-only supervision.

Component-comparison protocol. Full uses the same MCRA configuration as the main-results table: eight experts with Top-4 selection and the training settings in Appendices B.1–B.6. Component controls retain these settings except for the component being examined; T-Adapter follows its main-table protocol above. The fixed-scale operator has 67.6M trainable parameters, compared with 125.3M for MCRA, and evaluates a simpler context-adaptation pathway. TF-Mix instead retains the multi-scale expert pool, layer fusion, and final gate, replacing residue-dependent routing with learned task-shared weights. Both Full and TF-Mix select four of eight experts, so this com-

Table 11: Executed PEFT arguments. LoRA and DoRA differ only in the DoRA flag in this configuration.

Argument	LoRA	DoRA
`r`	720	720
`lora_alpha`	720	720
`lora_dropout`	0.05	0.05
`target_modules`	`[query, value]`	`[query, value]`
`bias`	`none`	`none`
`init_lora_weights`	`True`	`True`
`use_rslora`	`False`	`False`
`use_dora`	`False`	`True`
`inference_mode`	`False`	`False`

parison holds the available scale profiles and selection count fixed. Appendix B.11 gives the shared settings and the two allocation rules.

B.8 PEFT CONFIGURATION AND COMPARISON DESIGN

Table 3 compares supervised adaptation methods on Human-PPI, Anti. Res., and Sor. Sig. under a common ESM2 backbone. The scope is limited by the available compute budget; Appendix B.10 explains the biological and modeling questions covered by this subset.

The ESM2 wrapper implements LoRA and DoRA with PEFT. Both target the attention `query` and `value` linear modules, leaving the original backbone weights frozen; DoRA additionally enables `use_dora=True`. Both use rank $r = 720$, $\alpha = 720$, and dropout 0.05. Standard LoRA scaling is $\alpha/r = 1$, with rank-stabilized scaling disabled. The complete PEFT arguments are listed in Table 11. With the MCRA branch disabled, the adapted final-layer features pass through the shared $1280 \to 480$ projection and layer normalization before task prediction. The low-rank parameters and the downstream projection, normalization, and task head are trainable; DoRA also learns its magnitude parameters. MCRA instead trains its layer fusion, side projections, context experts, router, fusion gate, and downstream prediction modules while freezing the ESM2 backbone.

The trainable counts in Table 5 refer to the complete instantiated model, including downstream modules, rather than the low-rank matrices alone. The common optimization and validation rule is given in Appendix B.3; it is distinct from any method-specific hyperparameter search. Comparing the three methods at similar total trainable budgets addresses how that capacity is allocated between backbone updates and an external adaptation pathway. Rank 720 is a high-budget configuration relative to ESM2's hidden size 1280, not a test of low-rank efficiency at small parameter budgets. The results do not establish a ranking across ranks, target-module choices, or all 13 tasks.

B.9 FORWARD PROFILING AND PARAMETER ACCOUNTING

Measured region and timing statistics. The provided forward profiler selects a real batch from the task's test loader and reuses that batch across timing rounds. Data loading and model construction occur before timing. In online-backbone mode, each measured forward includes tokenization, host-to-device transfers, the ESM2 backbone, the adaptation and prediction modules, and the task loss returned by `UniModel.forward`; it does not reuse precomputed ESM2 representations. The model runs in evaluation mode under `torch.inference_mode()`.

The profiler defaults to bfloat16 autocasting, two warm-up forwards, and three timing rounds with five forwards per round. These are configurable profiler defaults. CUDA is synchronized before and after each timed round. The `latency_ms_per_sample` field is synchronized wall-clock duration divided by the number of forwards and the batch size; a separate CUDA-event field records device timing. Summaries use the arithmetic mean and sample standard deviation across rounds, rather than variation across training seeds. The JSON report records the effective arguments, task, batch index and size, sequence-length statistics, padding fraction, precision, device, checkpoint-

loading status, and individual rounds so that the measurement configuration remains distinguishable from the training configuration.

FLOPs and memory. FLOPs are counted once on the fixed model and batch using PyTorch's operator-level FLOP counter, with the PyTorch profiler as a fallback, and normalized by batch size. The same estimate is reused across timing rounds. This accounting follows the supported operators of the selected counter. Peak memory is CUDA peak allocated memory, expressed in GiB, and includes resident model parameters and forward activations. The profiler separately records reserved memory and the allocation increase above the post-warm-up baseline. Parameter counts, FLOPs, latency, and peak memory therefore describe different aspects of the forward pass.

Sparse execution and active parameters. The standard expert-mixture implementation evaluates all experts before Top-k weighted fusion. The separate sparse profiler provides `token_union`, `sequence`, and `batch` dispatch modes. In `token_union` mode, residue-wise Top-k selections are retained and their union is formed for each sequence. Each selected expert receives the complete sequence feature array for the samples routed to it, preserving its convolutional neighborhoods and full-sequence attention context. An expert is skipped for a sequence only when none of its positions selects that expert. Sequence- and batch-level dispatch instead select a shared expert subset at their respective aggregation levels and constitute distinct execution modes.

For a batch of B sequences, let $P_{\text{nonexpert}}$ count all model parameters outside the expert pool, P_e the number of parameters in expert e, and n_e the number of sequences dispatched to that expert. The sparse profiler defines

$$P_{\text{active/sample}} = P_{\text{nonexpert}} + \frac{1}{B}\sum_e n_e P_e, \qquad \text{Active } (\%) = 100\frac{P_{\text{active/sample}}}{P_{\text{total}}}. \tag{18}$$

This is a sequence-dispatch parameter accounting convention: all non-expert parameters are included, and expert parameters are weighted by the fraction of sequences dispatched to them. It is not a count of trainable parameters or a residue-wise k/E fraction of the full model. In particular, Top-4 at each residue can activate more than four experts for a sequence. The sparse report also records the dispatch mode, effective Top-k, expert count, and expert execution fraction. The distinction between these modes is necessary when relating the active-parameter column to the measured forward costs in Table 5. Top-4 routing alone therefore does not imply half the expert FLOPs or a particular latency reduction. Sequence- and batch-level dispatch alter the selection rule and should not be treated as equivalent to residue-wise routing without an output-equivalence check. The reported costs characterize the profiled implementation and workload, not an implementation-independent benefit of sparse routing.

B.10 Biological scope and interpretation of the focused comparisons

The 13-task study establishes the breadth of the main ESM2 evaluation. Two smaller comparisons ask complementary questions: whether a different adaptation mechanism helps when the backbone is held fixed, and whether the same adaptation architecture transfers when the backbone changes. Available compute limited both extensions to three tasks. The task sets are not random samples of the benchmark, and the coverage described below should be read as an interpretation of their scope rather than evidence that three tasks are sufficient for a general ranking.

Task-specific performance in the main benchmark. The main benchmark spans structural, functional, and protein-property tasks. MCRA does not exceed the strongest baseline on Human-PPI or DeepSoluE. A possible explanation for PPI is that it adapts individual protein representations without an explicit cross-protein interaction module. The DeepSoluE result also shows that the value of additional context adaptation depends on the prediction endpoint. Improvements on DeepSol, DL Bin, Clo. CLF, and Mat. Pro. are more modest than on Contact.

Why compare with LoRA and DoRA on these tasks? LoRA learns factorized updates to pretrained weights, whereas DoRA separately adapts weight magnitude and direction (Hu et al., 2022a; Liu et al., 2024). They provide relevant alternatives to spending trainable capacity on an external context-adaptation pathway. Protein-specific studies have demonstrated competitive PEFT on PPI

and broader transfer tasks (Sledzieski et al., 2024; Schmirler et al., 2024), making them substantive baselines rather than nominal parameter-count controls.

Human-PPI asks whether two proteins interact and therefore tests the use of paired protein representations, rather than a single-protein annotation. It also connects the comparison to a task family already studied with protein LoRA (Sledzieski et al., 2024). Anti. Res. asks for a protein's antibiotic resistance class, distinguishing 19 functional categories in the benchmark derived from CARD (Hu et al., 2022b). It probes multiclass functional discrimination, but the class label does not identify the causal residues or resistance mechanism. Sor. Sig. predicts nine types of sorting signals. These short sequence signals help govern cellular targeting, providing a biological setting in which local sequence patterns matter (Thumuluri et al., 2022). Our task predicts their presence at the protein level, not their residue boundaries. Together, the three tasks cover paired binary, single-protein multiclass, and multilabel outputs, with 30,133, 2,703, and 1,484 training examples, respectively. This contrasts relational, functional, and signal-related prediction under different amounts of supervision; it does not test all structural or biophysical endpoints.

What can explain the PEFT results? MCRA exposes intermediate-layer features directly to the task predictor and adds explicit local convolutional profiles before global attention. A plausible hypothesis is that these operations provide a useful inductive bias for combining localized sequence cues with protein-level context. This is a modeling interpretation, not a demonstrated cause of the gains. LoRA and DoRA also modify contextual representations through attention and are not confined to local or global features. Nor does DoRA's greater flexibility guarantee a higher score on every protein task: here it exceeds LoRA on Sor. Sig. but not on Human-PPI or Anti. Res.

Prior protein PEFT work found that adaptation targets and ranks matter, including a preference for key/value updates in its PPI setting rather than the query/value choice used here (Sledzieski et al., 2024). These findings place our rank-720 comparison in the context of protein-specific adaptation: the location and form of the trainable update matter alongside its size. Extending the comparison to other ranks and target modules would further characterize this design space.

Why evaluate backbone transfer on Contact, DeepSoluE, and Mat. Pro.? These tasks contrast a residue-pair structural target with two whole-protein properties. Contact retains pairwise spatial relationships instead of reducing the target to one protein label, as in structural transfer benchmarks such as TAPE (Rao et al., 2019). It therefore tests whether the adaptation pipeline can use residue-level representations from different encoders. DeepSoluE concerns soluble expression in *E. coli*; its original study combines physicochemical sequence features with learned representations (Wang & Zou, 2023). This provides a complementary whole-protein endpoint where local sequence patterns must be aggregated. Mat. Pro. concerns success in a protein-material production stage. It is not another contact or solubility label: the source study relates production outcomes to heterogeneous sequence-derived properties (Wang et al., 2014). In our experiments it is a binary classification task, not a quantitative yield-regression assay.

Across six additional PLMs, these three endpoints test whether the combined architecture remains useful as the pretrained representation changes. The baseline for each pair is the supervised Transformer adapter from the main table, not a head-only probe. Improvements in 17 of 18 pairs support transfer of the combined pipeline on this subset; the ProtST–DeepSoluE exception also shows that improvement is not universal. The results show that the same adaptation design can improve heterogeneous frozen representations, with benefits that depend on the backbone–task pair. Extending these comparisons to additional task families will test the broader portability of depth-and-scale adaptation.

B.11 Matched comparison of residue-wise and task-shared mixing

Task-specific depth and scale preferences leave open whether a task-wide expert mixture is sufficient. We compare residue-wise mixing (Full) with Task-Fixed-Multi (TF-Mix) on all 13 tasks in Table 4. Both retain the same eight multi-scale experts, learned fusion of the last 16 ESM2 layers, and final gate; TF-Mix learns one mixture shared by the task. Contact, Anti. Res., and DeepSol span residue-pair structure, functional classification, and protein-level solubility. The contrast tests input-dependent allocation while preserving task-specific depth selection and the expert pool.

Table 12: Shared Full–TF-Mix settings, consistent with the main-results MCRA protocol. The expert pool and Top-4 selection are retained; weights are residue-dependent or task-shared.

Component	Shared setting
Backbone	Frozen ESM2-650M
Layer fusion	Last 16 hidden layers; learned scalar mixing
Expert pool	8 experts, two per S/M/L/XL kernel profile in Table 8
Expert architecture	Hidden size 1920; 12 attention heads; dropout 0.1; five channel groups; learned γ_e initialized to zero
Fusion	Shared feature width 480; learned residue-wise frozen/adapted gate retained
Expert aggregation	Top-4 of 8 experts, followed by weight renormalization
Optimizer	AdamW; learning rate 10^{-4}; remaining settings as in Appendix B.3
Execution and batch size	Two GPUs; per-GPU batch: Contact 2, all other tasks 8; no gradient accumulation
Training and selection	Seed 2024; at most 50 epochs; early stopping patience 5; validation-based checkpoint selection
Objective	Same task losses and routing regularization as Appendix B.4: $\lambda_{\text{ent}} = 1$, $\lambda_{\text{bal}} = 10^{-2}$

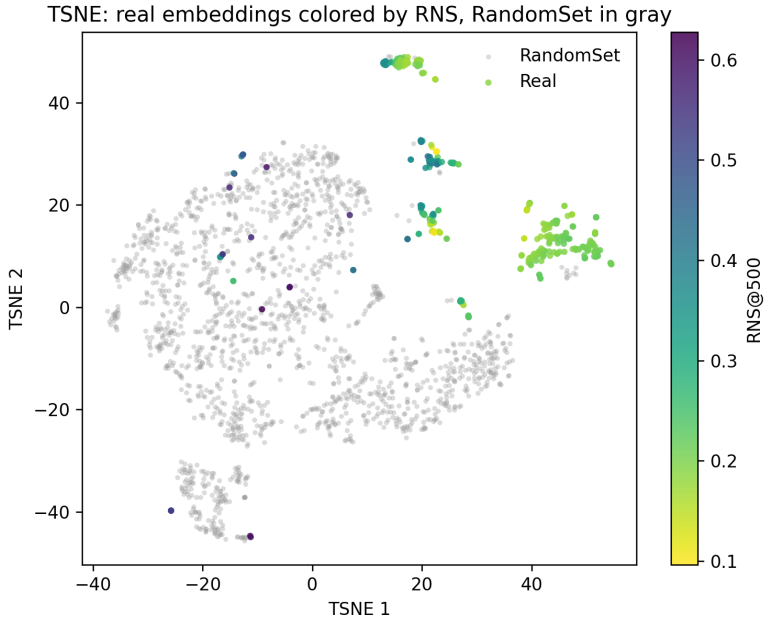
Residue-dependent versus task-shared weights. For task t, let X be the layer-fused input and $G_e(X)_i$ the output of expert e at residue i. The two mixtures are

$$M_i^{\text{res}}(x) = \sum_{e=1}^{8} p_{i,e}(x) G_e(X)_i,$$
$$M_i^{\text{task}}(x) = \sum_{e=1}^{8} w_{t,e} G_e(X)_i, \qquad \sum_{e=1}^{8} w_{t,e} = 1. \tag{19}$$

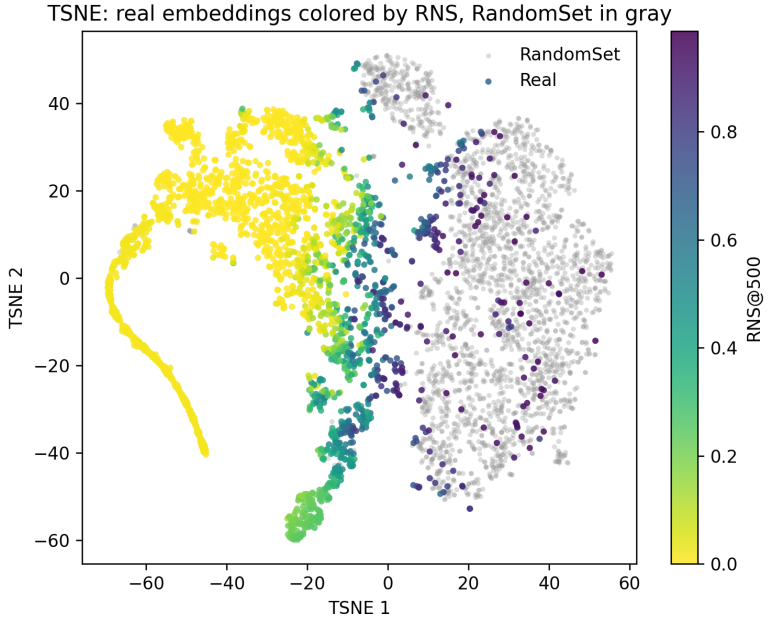
Full computes $p_i(x)$ with the original $1920 \to 1024 \to 8$ MLP and temperature $\tau = 1$, retaining and renormalizing the Top-4 weights at each residue, as in the main-results table. TF-Mix applies the same Top-4 selection and renormalization to a learned task-level probability vector to obtain w_t. The selected experts and weights are shared across all residues and proteins within a task at a given checkpoint, and are learned jointly with the other adaptation parameters. Thus, "fixed" means input-independent, not frozen during training. Expert outputs and the final gate remain input-dependent.

Both variants contain eight experts and select four for aggregation. Each expert processes the complete sequence, preserving convolutional and attention context. All four kernel profiles retain two independent experts each; "task-fixed" does not mean Small-only kernels or uniform weights. Unlike the single fixed-scale operator w/o Adapt., this contrast preserves the expert pool and the Top-4 setting of the main benchmark.

Evaluation and scope. Full and TF-Mix use the same held-out splits, task metrics, optimization schedule, effective batch size, random seed, and checkpoint-selection rule. Full has higher test scores on all 13 tasks; for Contact, Anti. Res., and DeepSol, the absolute differences are 0.0069, 0.0295, and 0.0276, respectively. With the multi-scale expert pool and learned depth fusion retained, these results support input-dependent allocation beyond learning a single task-wide mixture. Together with the layer preferences and final-layer comparison, they support the complementary roles of task-level depth selection and residue-wise context adaptation. The contrast concerns how expert allocation responds to the input: Full can change its mixture both across proteins and across residues, whereas TF-Mix applies one task-wide selection. This complements the depth analysis by connecting task-specific feature selection to input-specific context processing.

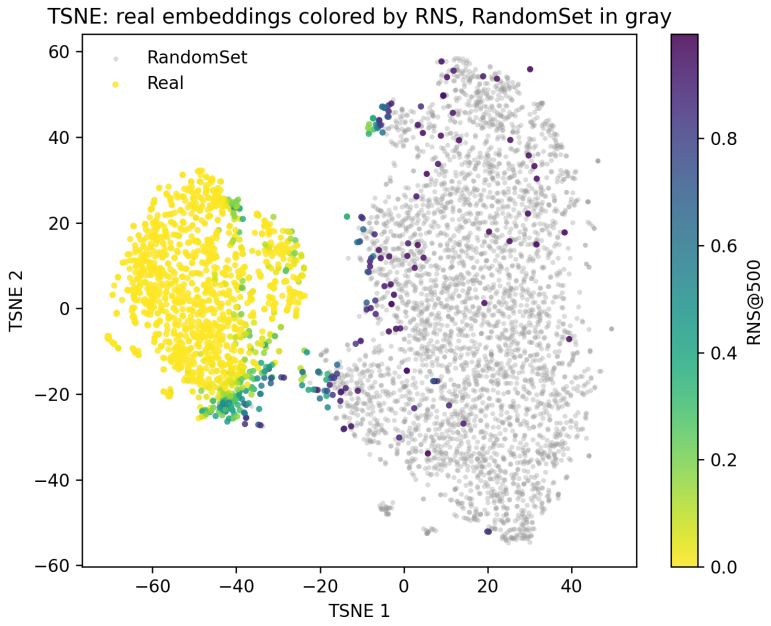


Anti. Res.

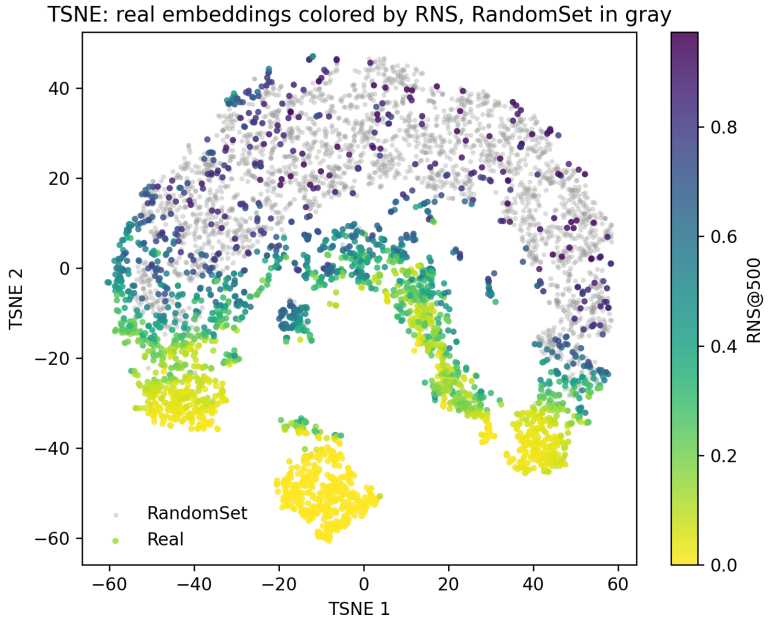


Clo. CLF

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

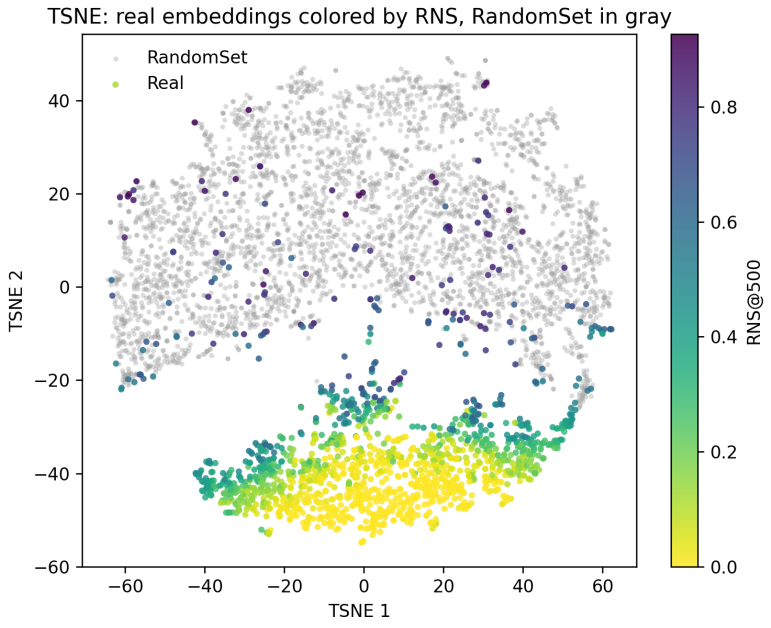


Contact

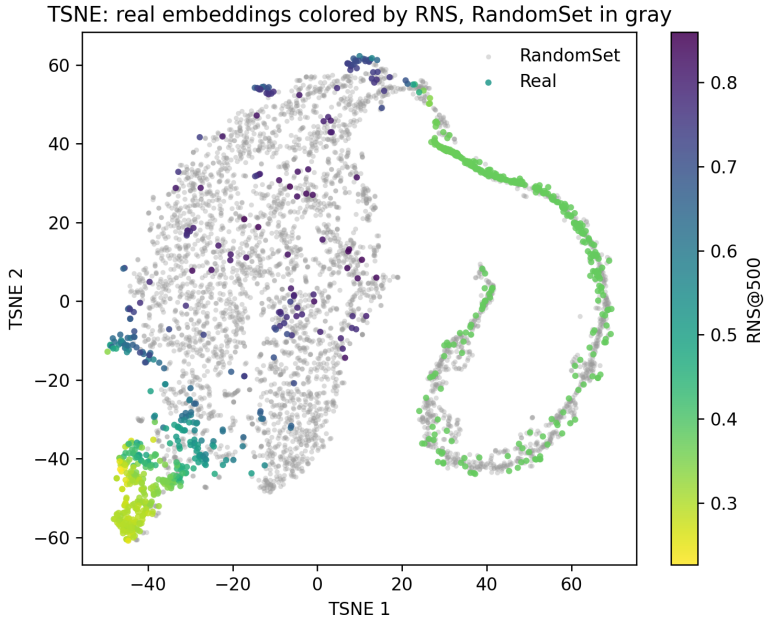


DeepSol

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

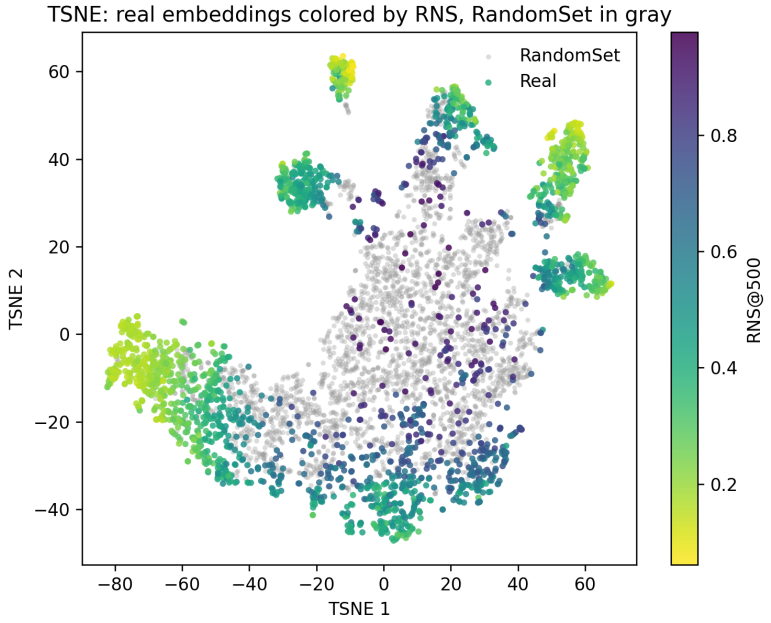


DeepSoluE

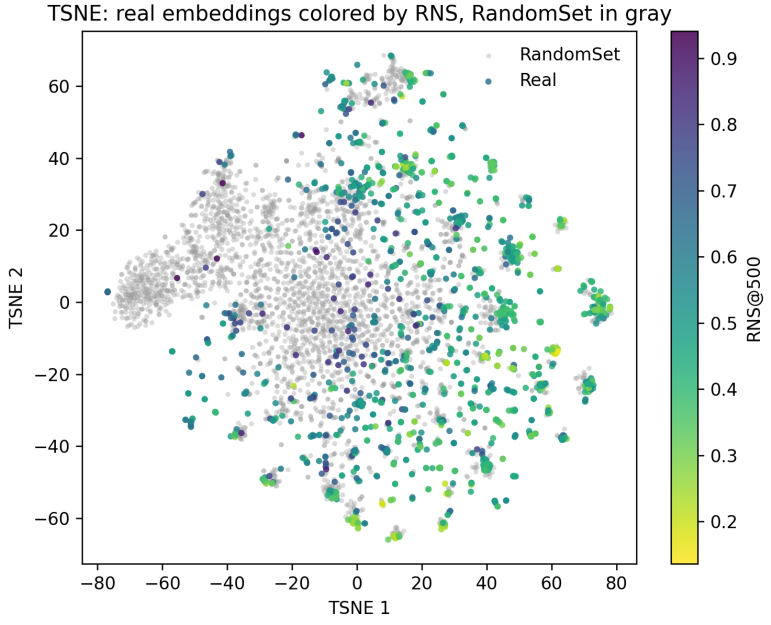


DL Bin

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

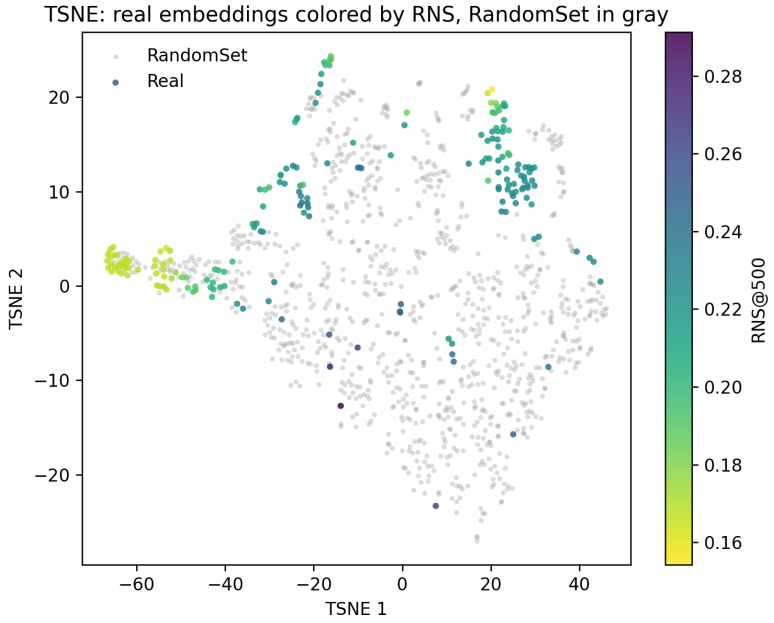


DL2 Multi

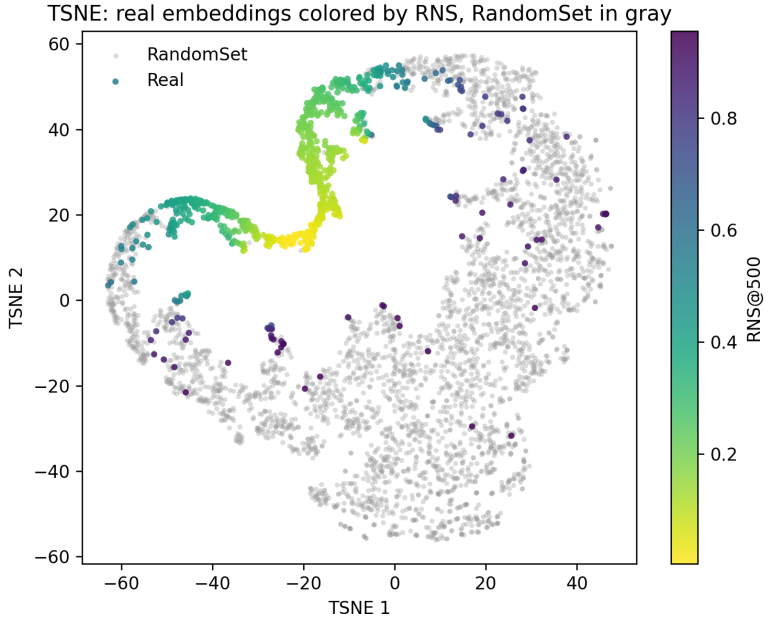


Fold Cls

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

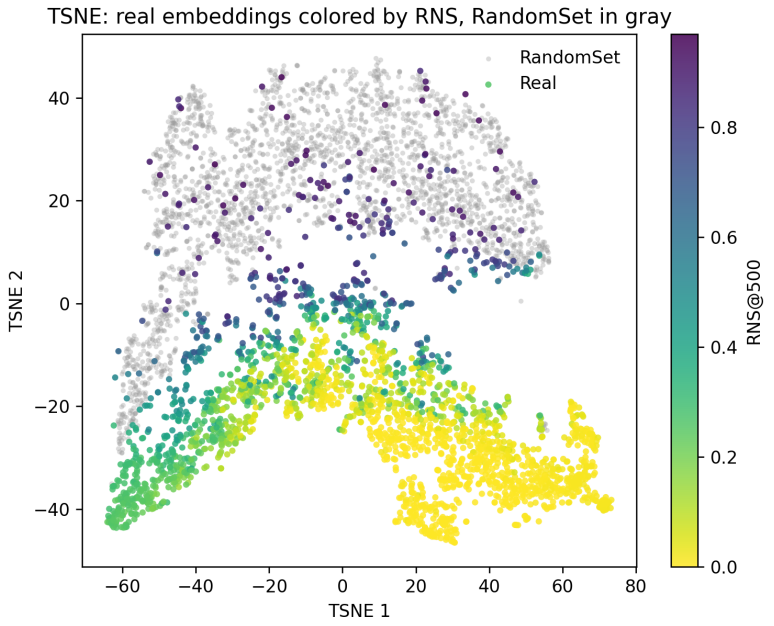


Human-PPI

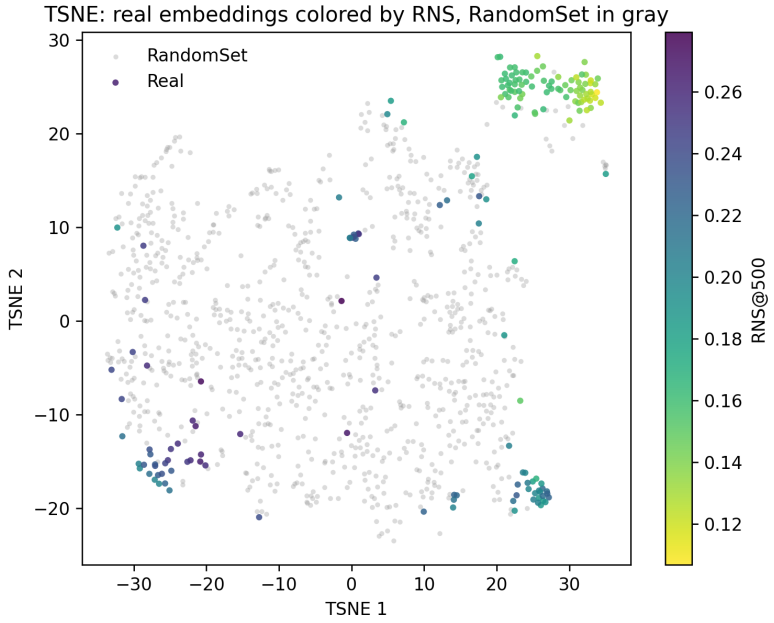


M. I. Bin.

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

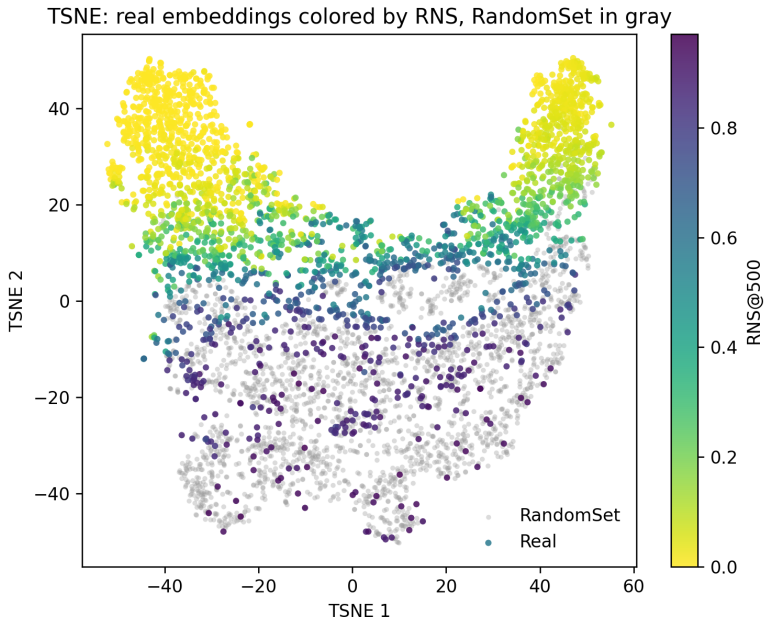


Mat. Pro.



Sor. Sig.

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.



Thermo.

Figure 8: Representation-space and shuffled-control analysis across all 13 tasks. Colored points denote real proteins and encode the residue neighborhood score (RNS), whereas gray points denote composition-preserving shuffled controls. Each task is projected independently; comparisons therefore concern real-versus-control organization within each panel rather than absolute coordinates across tasks. The task-dependent geometries provide a qualitative view of representations formed through adaptive depth fusion and scale-aware processing.

C PERTURBATION DIAGNOSTICS

Adaptive scale and depth selection should be responsive enough to register a controlled sequence change without becoming unnecessarily unstable. This appendix tests both sides of that behavior. Perturbation success measures whether MCRA detects the altered sequence at the prediction-score level; flip rate then tests whether that response overturns the discrete prediction; and scale-shift metrics test whether the receptive-field mixture is preserved. The fusion response additionally measures whether the balance between pretrained depth information and scale-adapted features remains stable. Together, these diagnostics connect predictive sensitivity to output and routing stability rather than treating routing weights as isolated visualizations.

We compare the model response on an original protein sequence and on synthetic mutant sequences generated from it. Each original sequence is treated as a wild type (WT) sequence $x = (a_1, \ldots, a_L)$, where a_i is the amino acid at residue position i and L is the sequence length. A mutant sequence is denoted $\tilde{x} = T_p(x)$, where T_p is a perturbation operator such as alanine scanning, physicochemical substitution, or sequence shuffling.

The analysis asks three linked questions. First, does the perturbation reduce the task-aware score for the original target, indicating that the model has detected the altered sequence? Second, when a discrete decision is defined, does that prediction remain unchanged? Third, does MCRA preserve its context-scale routing distribution? The perturbation experiment therefore separates score-level sensitivity from prediction and routing stability. It is a model diagnostic rather than an experimental mutation-effect assay; in particular, synthetic mutants are not interpreted as experimentally validated increases or decreases in protein function or structure.

C.1 SAMPLING PROTOCOL

Perturbation diagnostics use the frozen best-validation checkpoint for each task and held-out test examples only; no model parameter is updated. The analysis samples at most 150 WT examples per task after excluding sequences shorter than 8 residues. When fewer than 150 eligible examples are available, all are retained. Otherwise, examples are selected with seed 2024 and then restored to their original test-set order.

For each WT example, one mutant is generated for every applicable combination of perturbation operator and window length. Local and block-based settings use window lengths 15, 31, 63, 127, and 255; a requested length longer than a sequence segment is clipped to the segment length. Up to three non-overlapping high-saliency windows are modified simultaneously for each local mutant. Context shuffle protects one selected window and shuffles its remaining sequence background, whereas global shuffle produces one full-sequence mutant without a window-length sweep. For Human-PPI, both protein chains are perturbed under this pairing policy. Saliency probing uses random seed 2125, and final mutant generation uses seed 2041.

Every mutant is paired with its source WT example and evaluated against the same original task label. Mutants that are identical to WT after an attempted operation are discarded. This occurs, for example, when a selected charge-neutralization window contains no charged residues. Consequently, the number of retained WT–mutant pairs can be smaller than the theoretical maximum and differs across tasks and perturbation types. With nine window-based operators, five requested lengths, and one global-shuffle control, the maximum is 46 mutant attempts per WT, or 6900 attempts for a task with 150 WT examples.

C.2 PERTURBATION OPERATORS

Window-based perturbations are applied to selected residue windows, whereas global controls are applied to the full sequence. The perturbations cover local identity removal, physicochemical modification, local order disruption, and global sequence disruption.

Implementation details are as follows. Ala, Gly, and Pro scans replace all residues in the selected window by A, G, and P, respectively, using G for pre-existing A or P and A for pre-existing G to ensure an actual sequence change. Hydrophobic residues are defined as $\{\mathrm{A}, \mathrm{I}, \mathrm{L}, \mathrm{M}, \mathrm{F}, \mathrm{W}, \mathrm{V}, \mathrm{Y}\}$; in the hydro/polar swap, these residues are replaced by residues sampled from $\{\mathrm{S}, \mathrm{T}, \mathrm{N}, \mathrm{Q}, \mathrm{D}, \mathrm{E}, \mathrm{K}, \mathrm{R}\}$, whereas other residues in the window are replaced by hydrophobic

Table 13: Settings used for the perturbation diagnostics.

Setting	Value used in the analysis
Evaluation split	Held-out test partition
WT budget	Up to 150 eligible examples per task
Minimum sequence length	8 residues
Base random seed	2024
Mutant replicates	1 per WT/operator/window setting
Window lengths	15, 31, 63, 127, 255 residues
Windows per local mutant	Up to 3 non-overlapping windows
Window selection	Model-saliency, alanine-window probe
Saliency stride	0.5 times the window length
Maximum saliency candidates	32 per sequence segment and window length
Human-PPI policy	Perturb both chains
No-change attempts	Discarded

Table 14: Perturbation operators used for model-sensitivity and routing-stability diagnostics.

Perturbation	Operation	Diagnostic role
Ala scan	Replace residues in selected windows by alanine	Removes side-chain identity while retaining a simple backbone-compatible residue
Gly scan	Replace residues in selected windows by glycine	Increases local flexibility and weakens side-chain constraints
Pro scan	Replace residues in selected windows by proline	Disrupts local secondary-structure propensity
Hydro/polar	Swap hydrophobic and polar/charged residue classes in selected windows	Tests sensitivity to local physicochemical character
Hydro patch	Replace selected windows with hydrophobic residues	Tests whether hydrophobic patches alter prediction and routing
Charge neut.	Replace charged residues by neutral residues where applicable	Tests sensitivity to charge information
Local shuffle	Shuffle residues within selected windows	Preserves local composition but disrupts local order
Context shuffle	Preserve selected windows and shuffle the outside context	Tests dependence on sequence background outside the local site
Block shuffle	Shuffle sequence blocks	Preserves local segments but disrupts long-range ordering
Global shuffle	Shuffle the full sequence	Composition-preserving global disruption control

residues. Hydrophobic patches sample from $\{\mathrm{L}, \mathrm{V}, \mathrm{I}, \mathrm{F}, \mathrm{M}, \mathrm{W}, \mathrm{Y}\}$. Charge neutralization maps acidic residues D/E to N/Q and basic residues K/R/H to Q/N/N, leaving uncharged residues unchanged. Shuffle-based perturbations preserve amino-acid composition within the shuffled region. Block shuffle partitions a sequence into contiguous blocks of the specified window length and permutes their order; global shuffle permutes the complete chain. A shuffle is retried up to ten times when it reproduces the original order, after which a deterministic reversal is used as a fallback. The operators require only sequence positions and do not use functional-site, DSSP, RSA, or other biological annotations.

C.3 Model-saliency window selection

For local perturbations, we select windows using a model-guided saliency scan. For a candidate window $w = (i, j)$ and a probe perturbation T_{probe} (alanine scanning by default), we compute a probe score drop

$$\Delta s_{\mathrm{probe}}(w) = s(x, y) - s(T_{\mathrm{probe}}(x; w), y), \tag{20}$$

where $s(\cdot, y)$ is the task-aware target score defined in Appendix C.4. For a window of length $|w|$, candidate starts are separated by approximately $0.5|w|$ residues, with the final sequence-aligned window included. At most 32 candidates are retained per sequence segment and window length; when more are available, they are sampled at evenly spaced start indices. Candidates are scored by $\Delta s_{\mathrm{probe}}(w)$ and ranked separately for each WT example, chain, and window length. The final mutant uses the three highest-ranked mutually non-overlapping windows whenever the sequence is long enough. If three non-overlapping windows cannot be obtained, the available high-ranked windows are used. Controlled random windows are used only when no saliency-ranked window is available for that sequence segment and length.

This selection deliberately targets regions to which the trained model is most sensitive. Because the same original target y is used to rank the alanine probe score drop and to evaluate the final perturbation response, the procedure is a label-aware, adversarial stress test rather than an unbiased estimate of robustness to uniformly random mutations. Window selection occurs only after training, does not update model parameters, and does not affect checkpoint selection. A random-window mode is implemented as a control, but the task-level results in this appendix use model-saliency selection.

C.4 Task-aware perturbation score

Let f_θ denote the trained model with parameters θ. For a wild-type–mutant pair $(x, \tilde{x})$, x denotes the original WT sequence and $\tilde{x}$ denotes the perturbed mutant sequence. The label or target associated with the WT example is denoted by y. We define a task-aware target score $s(x, y)$ so that larger values always mean stronger model support for the original target y. The perturbation effect is then measured as

$$\Delta s = s(x, y) - s(\tilde{x}, y). \tag{21}$$

Here, $s(x, y)$ is the target score of the WT sequence and $s(\tilde{x}, y)$ is the target score of the mutant sequence evaluated against the same target y. Therefore, Δs measures how much the perturbation reduces the model's support for the original target. A positive Δs means that the mutant receives a lower target score than the WT sequence, and is counted as a successful perturbation. The perturbation success rate is

$$\mathrm{SuccessRate} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{1}\left[\Delta s_n > 0\right]. \tag{22}$$

In this equation, N is the number of WT–mutant pairs, Δs_n is the score drop for the n-th pair, and $\mathbf{1}[\cdot]$ is an indicator function that equals 1 when the condition is true and 0 otherwise. A high success rate therefore means that the model consistently registers the targeted sequence perturbation as reduced support for the original WT target. It is a score-level sensitivity measure; prediction and routing stability are assessed separately below.

The score $s(x, y)$ is chosen according to task type, because different tasks have different output formats. For multiclass classification, let $z_c(x)$ be the model logit for class c and let y be the true class. We use the true-class margin

$$s(x, y) = z_y(x) - \max_{c \neq y} z_c(x). \tag{23}$$

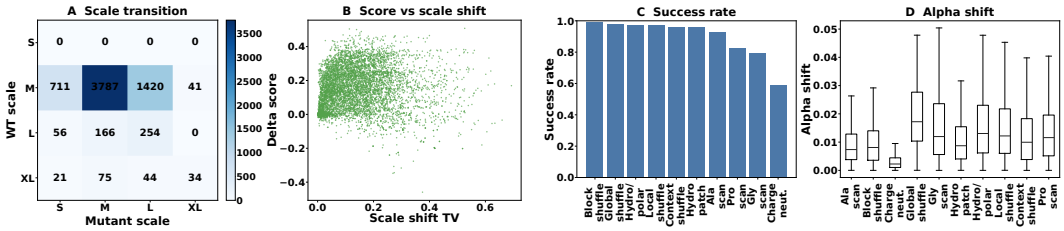


Figure 9: Contact perturbation diagnostics: (A) WT-to-mutant scale transitions, (B) score change versus scale-shift TV, (C) success rates, and (D) ESM2–MCRA fusion-gate shifts.

Here, $z_y(x)$ is the logit of the correct class and $\max_{c \neq y} z_c(x)$ is the largest competing logit among all incorrect classes. A larger margin indicates stronger preference for the correct class.

For binary classification, let $z(x)$ be the model logit and $y \in \{0, 1\}$ be the binary label. We use

$$s(x, y) = (2y - 1)z(x). \tag{24}$$

This makes the score direction consistent for both labels: when $y = 1$, $s(x, y) = z(x)$, and when $y = 0$, $s(x, y) = -z(x)$. Thus, larger $s(x, y)$ always means stronger support for the true binary label.

For multilabel classification, let $\mathcal{Y}^+$ be the set of positive labels for the sample and let $z_k(x)$ be the logit for label k. We define

$$s(x, y) = \frac{1}{|\mathcal{Y}^+|} \sum_{k \in \mathcal{Y}^+} z_k(x). \tag{25}$$

Here, $|\mathcal{Y}^+|$ is the number of positive labels. This score measures the average model support for the labels that should be active. If a task example has no positive label, the implementation instead uses the mean negative logit, $-K^{-1} \sum_{k=1}^{K} z_k(x)$, so that a larger score still means stronger support for the all-negative target.

For Contact, the output is a residue-pair contact map rather than a class logit. We define $s(x, y)$ as the average model probability assigned to native contact pairs in the WT contact label y. This measures whether the perturbation weakens the model's original native-contact prediction. It should be interpreted as a sensitivity diagnostic, not as mutant structural accuracy.

The numerical scale of s differs by task: multiclass tasks use a logit margin, binary and multilabel tasks use logits, and Contact uses a probability. Thus, Δs is interpretable within a task but its absolute magnitude should not be compared directly across tasks. Likewise, SuccessRate is not a WT-versus mutant classification accuracy. It is the fraction of synthetic mutants for which the trained model gives less support to the original WT target.

For classification tasks, the flip rate is the fraction of pairs for which the predicted class changes after perturbation. Binary decisions use a sigmoid threshold of 0.5; for multilabel tasks, a flip means that the thresholded set of predicted labels changes in any position. Flip rate is undefined for Contact and for regression because these outputs do not have a single comparable class decision.

These values measure model sensitivity only and should not be interpreted as true increases or decreases in the underlying biological property. None of the 13 tasks summarized in Table 16 is evaluated as regression; Thermo., DeepSol, and DeepSoluE use binary classification with AUROC.

C.5 Scale-routing and fusion diagnostics

Figure 9 brings these diagnostics together for Contact. The transition matrix tracks the dominant sequence-averaged scale, the scatter plot relates score sensitivity to scale redistribution, and the remaining panels show perturbation success and fusion-gate responses. These complementary views distinguish changes in the prediction from changes in the aggregate routing distribution.

This section describes how to read the routing diagnostics. MCRA assigns each valid residue to one or more of 8 context experts. Let $r_{i,e}(x)$ be the normalized Top-4 routing probability of expert

Table 15: Perturbation diagnostic metrics used in the analysis.

Metric	Meaning
`delta_score`	Task-aware target-score decrease, $\Delta s = s(x, y) - s(\tilde{x}, y)$
`success_rate`	Fraction of mutant pairs with $\Delta s > 0$
`flip_rate`	Fraction of classification pairs whose predicted class or label changes after perturbation
`dominant_scale_change_rate`	Fraction of pairs whose most-used scale group changes after perturbation
`mean_scale_shift_tv`	Mean global total-variation distance between wild-type and mutant scale distributions
`mean_alpha_shift_global`	Mean D_β; legacy field name for the absolute change in the ESM2–MCRA fusion gate
`dominant_scale_transition`	Raw counts of WT dominant scale versus mutant dominant scale, visualized as a transition heatmap

e at valid residue i, and let $\mathcal{E}_g$ be the two experts assigned to scale group $g \in \mathcal{G}$, where $\mathcal{G} = \{\text{Small}, \text{Medium}, \text{Large}, \text{XLarge}\}$. For a sequence with L_x valid residues, the sequence-level scale distribution is

$$p_g(x) = \frac{1}{L_x} \sum_{i=1}^{L_x} \sum_{e \in \mathcal{E}_g} r_{i,e}(x), \qquad \sum_{g \in \mathcal{G}} p_g(x) = 1. \tag{26}$$

Padding and sequence-boundary tokens are excluded. The global scale-shift between WT and mutant is the total-variation (TV) distance

$$D_{\text{scale}}(x, \tilde{x}) = \frac{1}{2} \sum_{g \in \mathcal{G}} |p_g(x) - p_g(\tilde{x})| \,. \tag{27}$$

This quantity lies in $[0, 1]$: 0 indicates identical sequence-averaged scale distributions, while 1 indicates non-overlapping distributions. Averaging can cancel changes at different residues, and summing experts within a scale can hide exchanges between the two experts in that group. A small global TV is therefore not evidence that individual residues keep their routing assignments. The dominant scale is $g^*(x) = \arg\max_g p_g(x)$, and the dominant-scale change indicator is

$$\mathbf{1}\left[\arg\max_g p_g(x) \neq \arg\max_g p_g(\tilde{x})\right]. \tag{28}$$

MCRA also uses a fusion gate to combine pretrained ESM2 features with scale-adapted features. In the main-paper equations this gate is denoted $\beta_i(x)$, where larger values favor frozen ESM2 features and smaller values favor MCRA features. Let $\bar{\beta}(x) = L_x^{-1} \sum_i \beta_i(x)$ be its mean over valid residues. The global fusion response is

$$D_\beta(x, \tilde{x}) = \left|\bar{\beta}(x) - \bar{\beta}(\tilde{x})\right|. \tag{29}$$

A large Δs together with a small D_{scale} indicates a prediction-score response with little change in aggregate scale mass. A small D_β describes a similar sequence-averaged gate, not necessarily unchanged residue-level gating or equal contributions from the two branches.

C.6 Perturbation summary metrics

Table 15 summarizes the task-level perturbation metrics used to compare prediction sensitivity and routing stability across protein tasks. We keep the metrics that correspond directly to the task-wise figures in Appendix C.8 and the task-level summary table: prediction-score change, perturbation success rate, prediction flipping, global scale redistribution, dominant-scale changes, and fusion-gate response.

For N retained pairs, the task-level Scale TV, dominant-scale change rate, and gate shift are arithmetic means of D_{scale}, the dominant-change indicator, and D_β, respectively. The overall values are *pair-weighted*: local and block operators contribute one bucket for each of five window lengths,

Table 16: Task-level perturbation diagnostics. Values are pair-weighted means; Δs is comparable within, but not across, tasks. Succ. is the fraction of pairs with reduced support for the original WT target, whereas Flip is the fraction whose predicted class or label changes. A dash indicates that flip rate is not applicable to Contact.

Task	Δs	Succ.	Flip	Dom. change	Scale TV	β shift
DeepSol	1.4561	0.7400	0.4382	0.0000	0.0000	0.0766
Fold Cls	8.1189	0.7581	0.6730	0.1043	0.1336	0.1295
Contact	0.1510	0.8880	–	0.3834	0.1413	0.0119
Anti. Res.	4.5119	0.7534	0.4909	0.0000	0.0056	0.0329
Human-PPI	0.9594	0.6004	0.1188	0.0000	0.0000	0.0000
Thermo.	3.2817	0.7251	0.2962	0.0000	0.0224	0.0000
Sor. Sig.	3.1182	0.7840	0.5903	0.0000	0.0025	0.0573
M. I. Bin.	0.5969	0.6211	0.2290	0.0000	0.0003	0.1115
DL Bin	1.6256	0.6109	0.2049	0.0000	0.0000	0.1779
DL2 Multi	1.3577	0.7188	0.5573	0.0092	0.0357	0.0449
Clo. CLF	1.2229	0.6777	0.3388	0.0000	0.0052	0.2252
DeepSoluE	0.3852	0.6240	0.3112	0.0052	0.0224	0.0388
Mat. Pro.	0.2031	0.5146	0.2222	0.0000	0.0000	0.0334

whereas global shuffle contributes one bucket. The overall mean is therefore influenced more strongly by window-based operators; the perturbation-specific panels in Appendix C.8 should be used when comparing operators. The transition heatmap contains counts rather than row-normalized probabilities, so darker cells indicate more observed pairs, not necessarily a higher conditional transition probability.

Multiple mutants derived from the same WT sequence are statistically dependent. These summaries are descriptive pair-level means and do not treat all pairs as independent biological replicates; no hypothesis test or confidence interval is attached to these diagnostic values.

Joint interpretation of sensitivity and stability. The positive mean score decreases and success rates above 0.5 show that targeted perturbations usually lower support for the original label. This is a prediction-level response; the perturbed windows are selected using the model itself, so the result is conditional on that selection procedure.

Among the 12 tasks for which a discrete flip is defined, nine have flip rates below 0.5. The unweighted mean flip rate is 0.3726, corresponding to a mean unchanged-prediction fraction of 0.6274, and the median flip rate is 0.3250. Nine tasks have a reported dominant-scale change rate of 0.0000, and 11 of the 13 tasks have mean Scale TV no greater than 0.0357. These values describe aggregate invariance, which is compatible both with stable adaptive routing and with a largely fixed task-conditioned mixture. They do not distinguish these explanations without router controls and residue-level change measurements.

The stability is task dependent rather than absolute. Human-PPI has the lowest flip rate (0.1188), followed by DL Bin (0.2049) and Mat. Pro. (0.2222), whereas Fold Cls (0.6730), Sor. Sig. (0.5903), and DL2 Multi (0.5573) change their discrete predictions more often than not. Fold Cls and Contact also show the largest scale redistributions; Contact is omitted from the flip-rate comparison because a class-level flip is not applicable. These exceptions identify tasks with a larger measured response. Contact's dominant-scale retention is 61.66% overall, the complement of its 38.34% change rate. Because the transition plot contains raw counts, its prominent Medium-to-Medium block does not imply uniformly high retention for all four starting scales; row-normalized rates are needed for that comparison.

C.7 Interpretation boundaries

These perturbation analyses are intended as model diagnostics rather than experimental mutation-effect measurements. The Thermo., DeepSol, and DeepSoluE tasks are classification tasks; if the same framework is later applied to a genuinely continuous property, predicted changes without

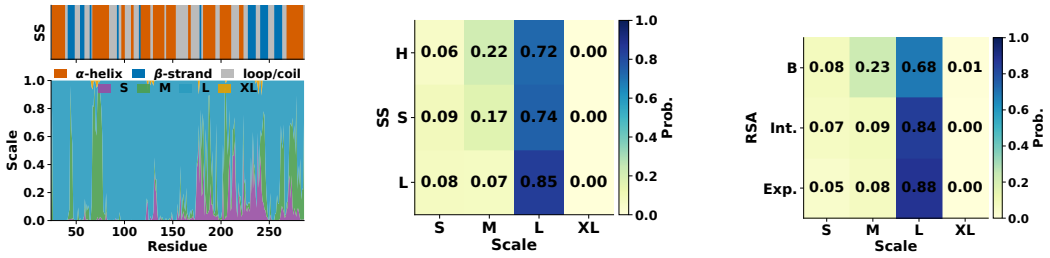


Figure 10: TEM-1 routing in structural context. Left: residue-wise Small (S), Medium (M), Large (L), and XLarge (XL) routing probabilities aligned with DSSP secondary structure. Middle: mean scale probabilities for α-helices, β-strands, and loop/coil residues. Right: mean probabilities for buried, intermediate, and exposed residues defined by relative solvent accessibility.

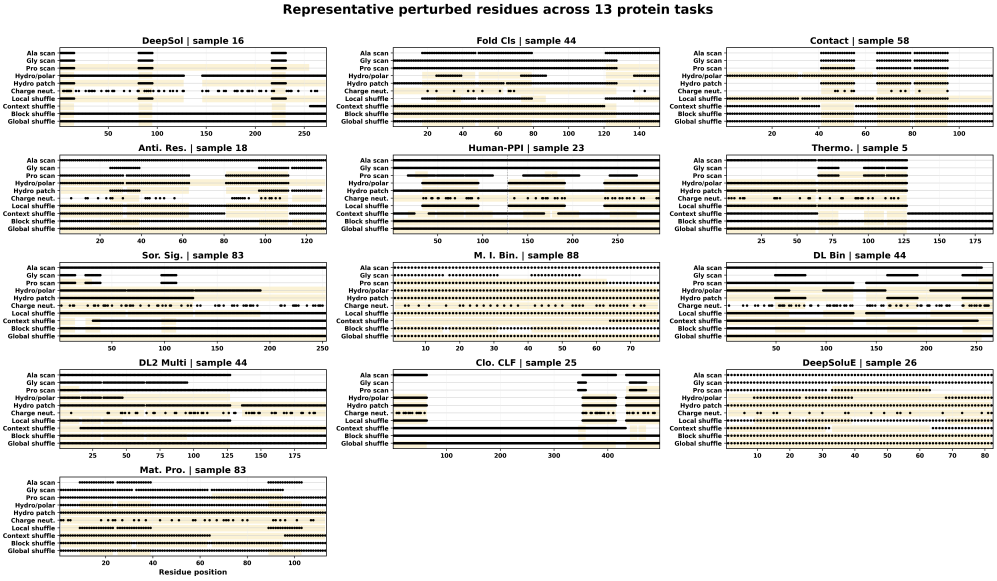


Figure 11: Representative perturbation sites across all protein tasks. Rows denote perturbation instances, shaded bands indicate selected windows, and black dots mark residues whose identities differ from WT.

mutant ground truth would indicate model sensitivity only. For Contact, mutant contact scores are evaluated against the native WT contact labels to measure disruption of the original prediction. They are not mutant structural accuracies because the synthetic mutants have no experimentally determined contact maps.

A high perturbation success rate establishes that the synthetic mutation is recognized at the score level; it is the sensitivity component of the analysis, not a stability metric and not a separately trained ability to classify natural versus mutant proteins. Low flip rates describe preserved decisions, whereas low Scale TV describes preserved sequence-averaged routing mass. Neither proves preservation of residue-level routing or physical biological structure. Random-window and untrained-router controls would help separate sensitivity induced by the selection procedure from behavior specific to learned context adaptation. Direct biological mutation-effect claims would require mutant labels such as deep-mutational-scanning measurements.

C.8 TASK-WISE PERTURBATION FIGURES

For each task, four complementary perturbation diagnostic plots are arranged in task-wise figures. Figures 12–18 show the four scalar diagnostic panels for each task: (i) the distribution of target-score decreases $\Delta s = s(x, y) - s(\tilde{x}, y)$ across perturbation types, where larger positive values indicate stronger degradation of the wild-type target score; (ii) the perturbation success rate, defined as the fraction of mutants with $\Delta s > 0$; (iii) the global scale-shift total variation distance, which measures how much the scale-routing distribution changes after perturbation; and (iv) the fusion-alpha shift, which measures the absolute change in the fusion gate between pretrained ESM2 features and scale-adapted MCRA features.

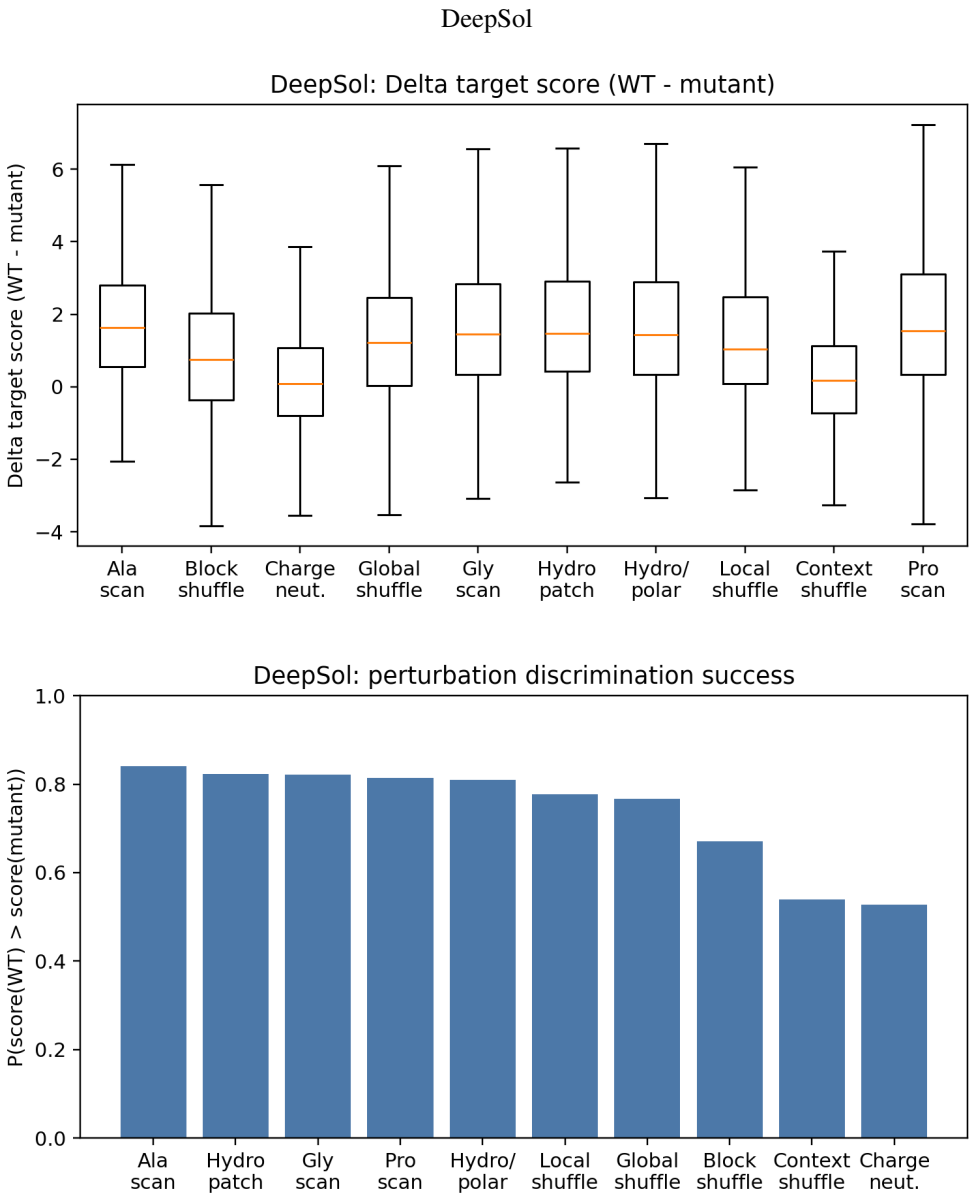


Figure 12: Scalar perturbation diagnostics for DeepSol and Fold Cls. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

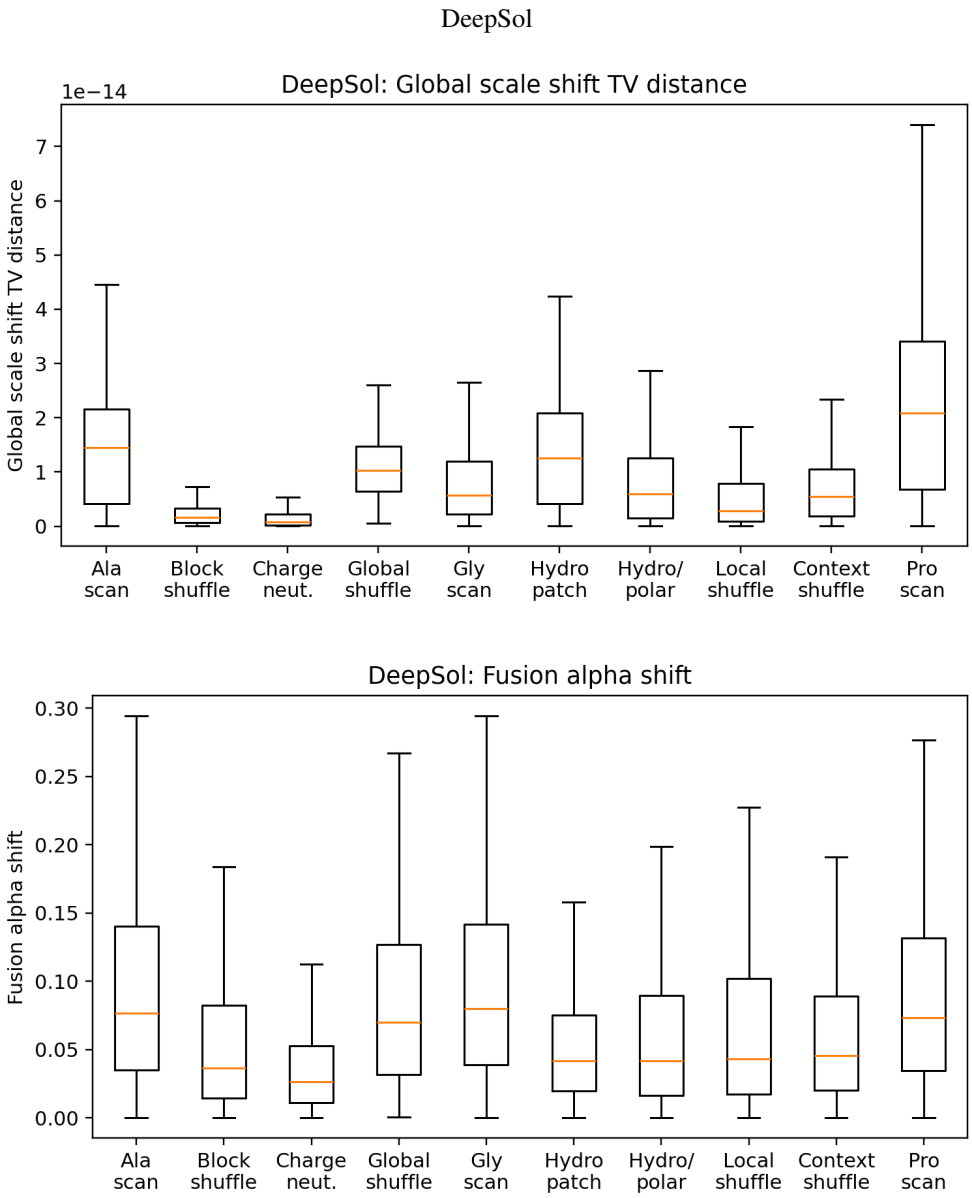


Figure 12: Scalar perturbation diagnostics for DeepSol and Fold Cls. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

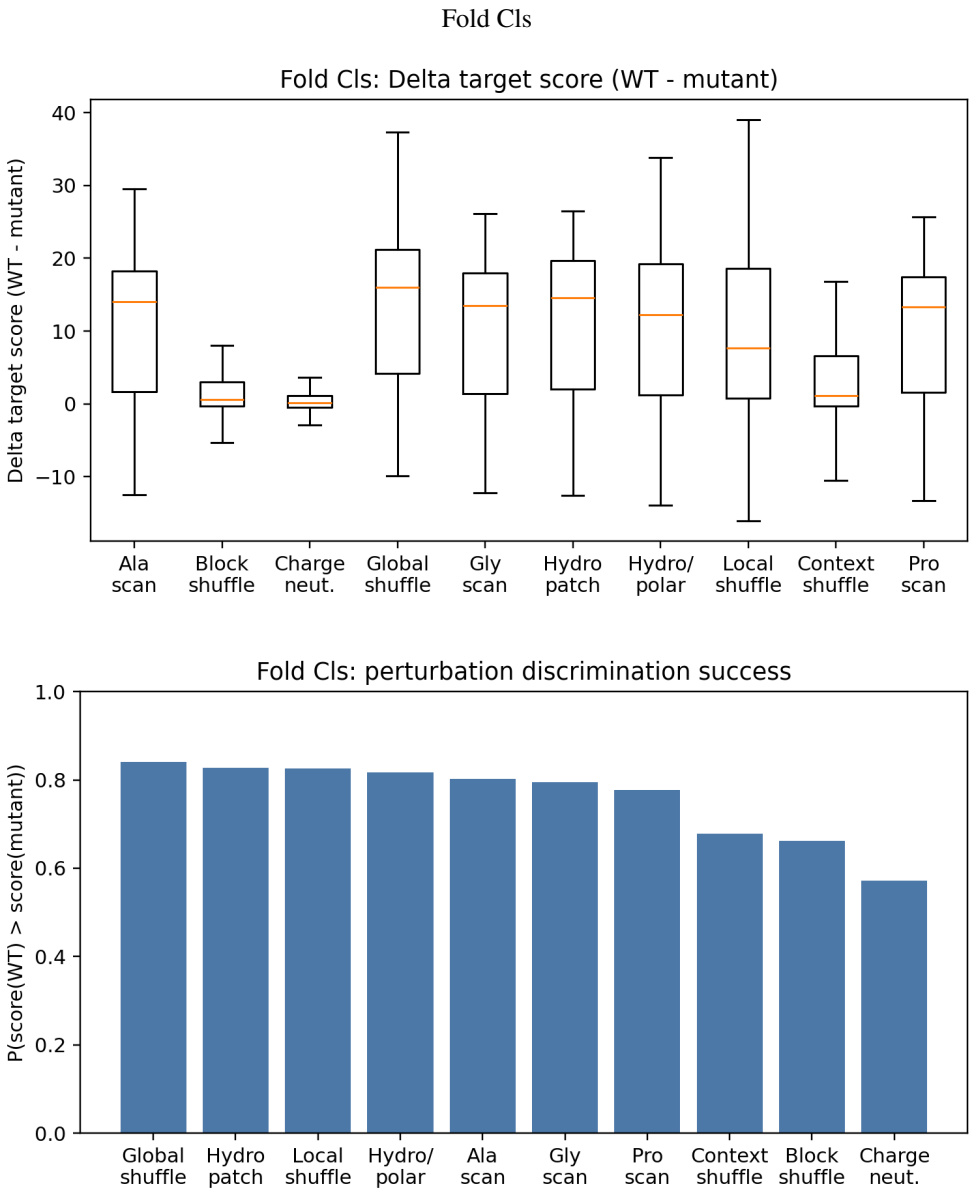


Figure 12: Scalar perturbation diagnostics for DeepSol and Fold Cls. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

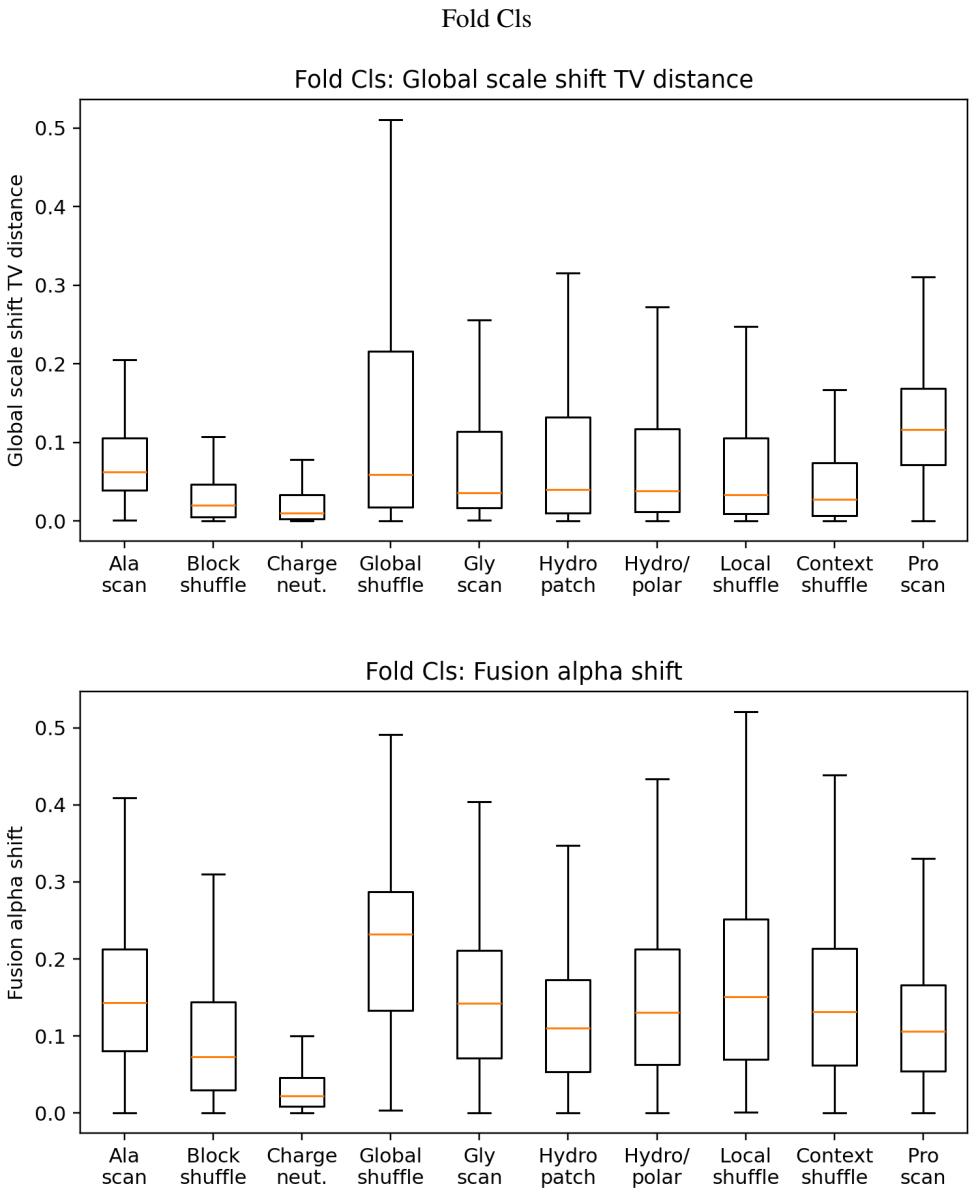


Figure 12: Scalar perturbation diagnostics for DeepSol and Fold Cls. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

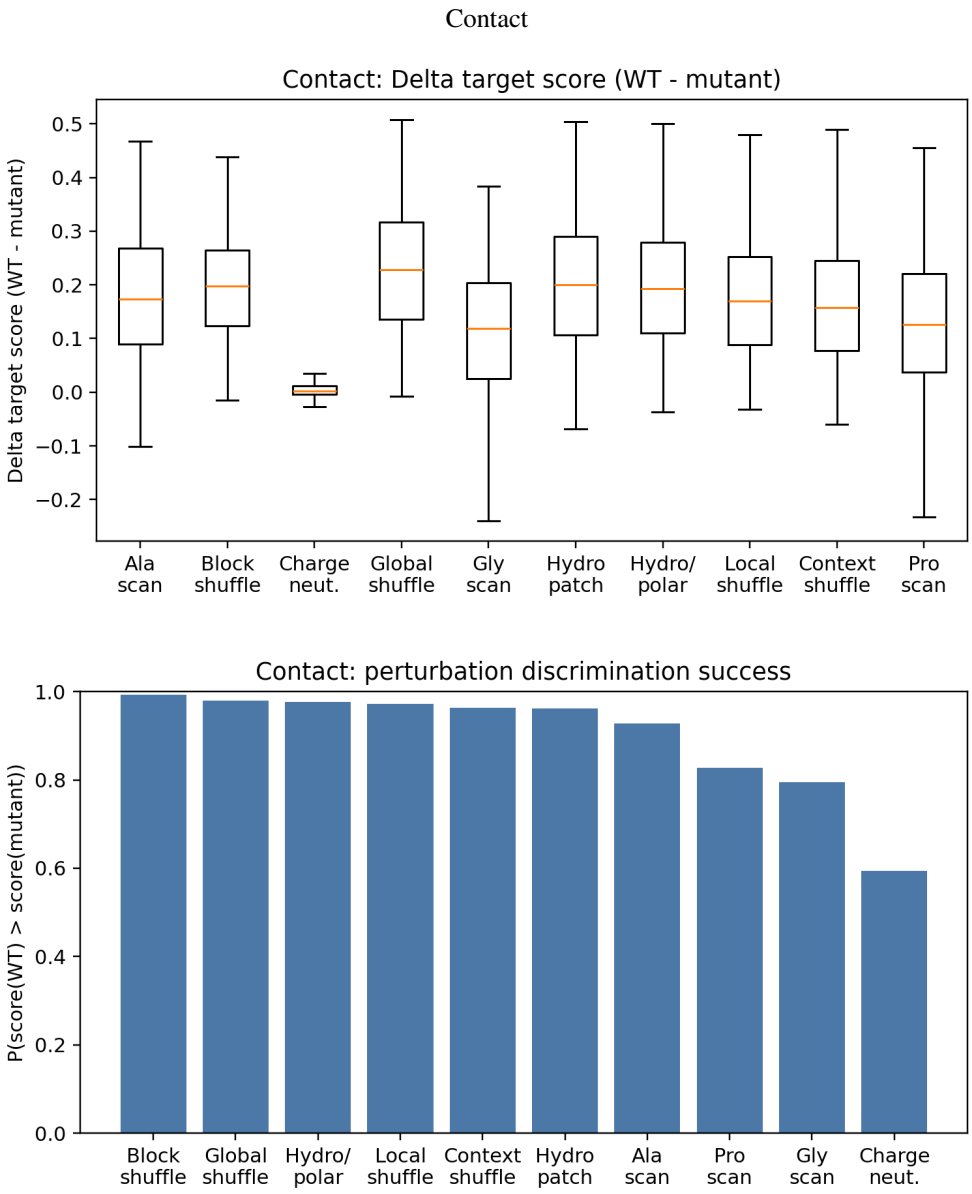


Figure 13: Scalar perturbation diagnostics for Contact and Anti. Res.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

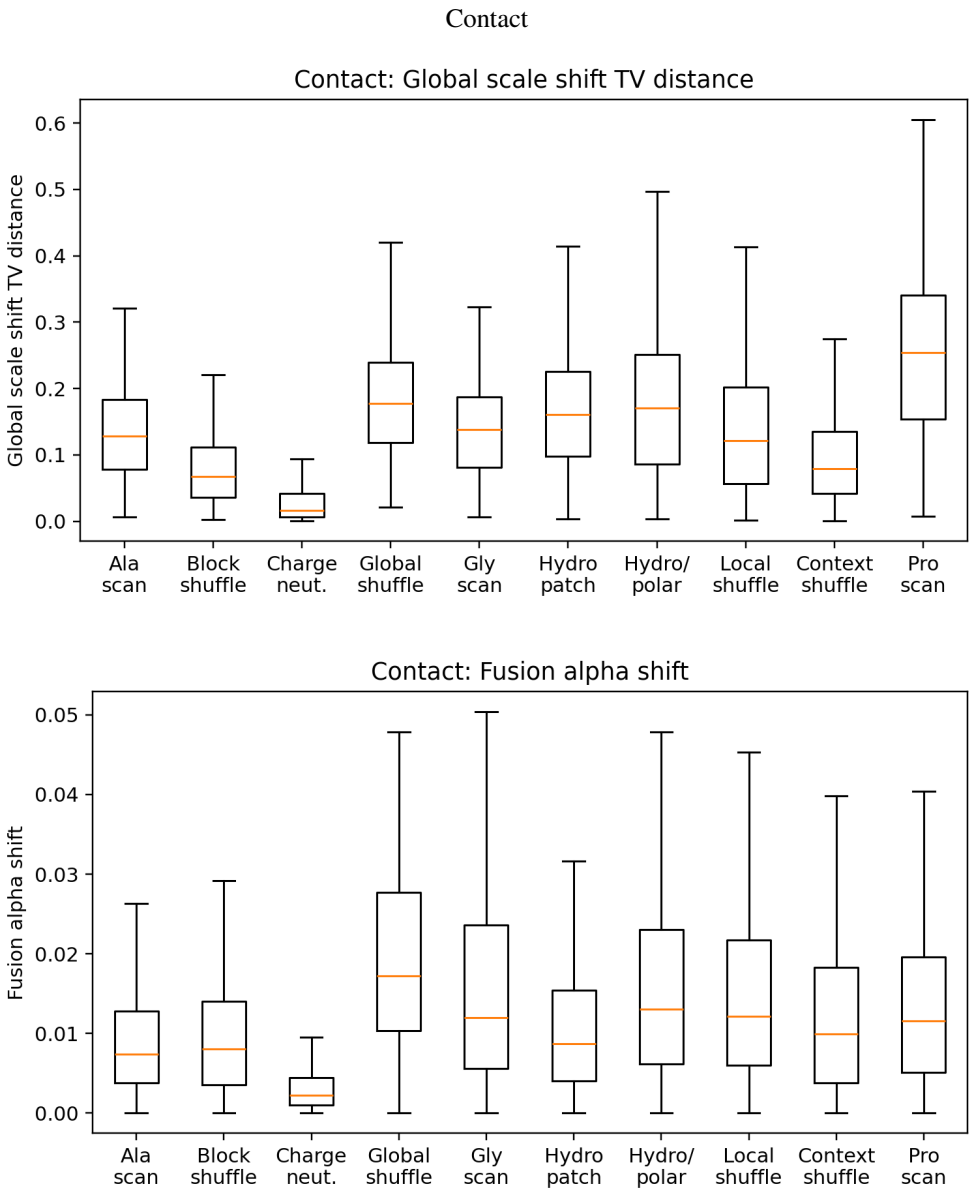


Figure 13: Scalar perturbation diagnostics for Contact and Anti. Res.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Anti. Res.

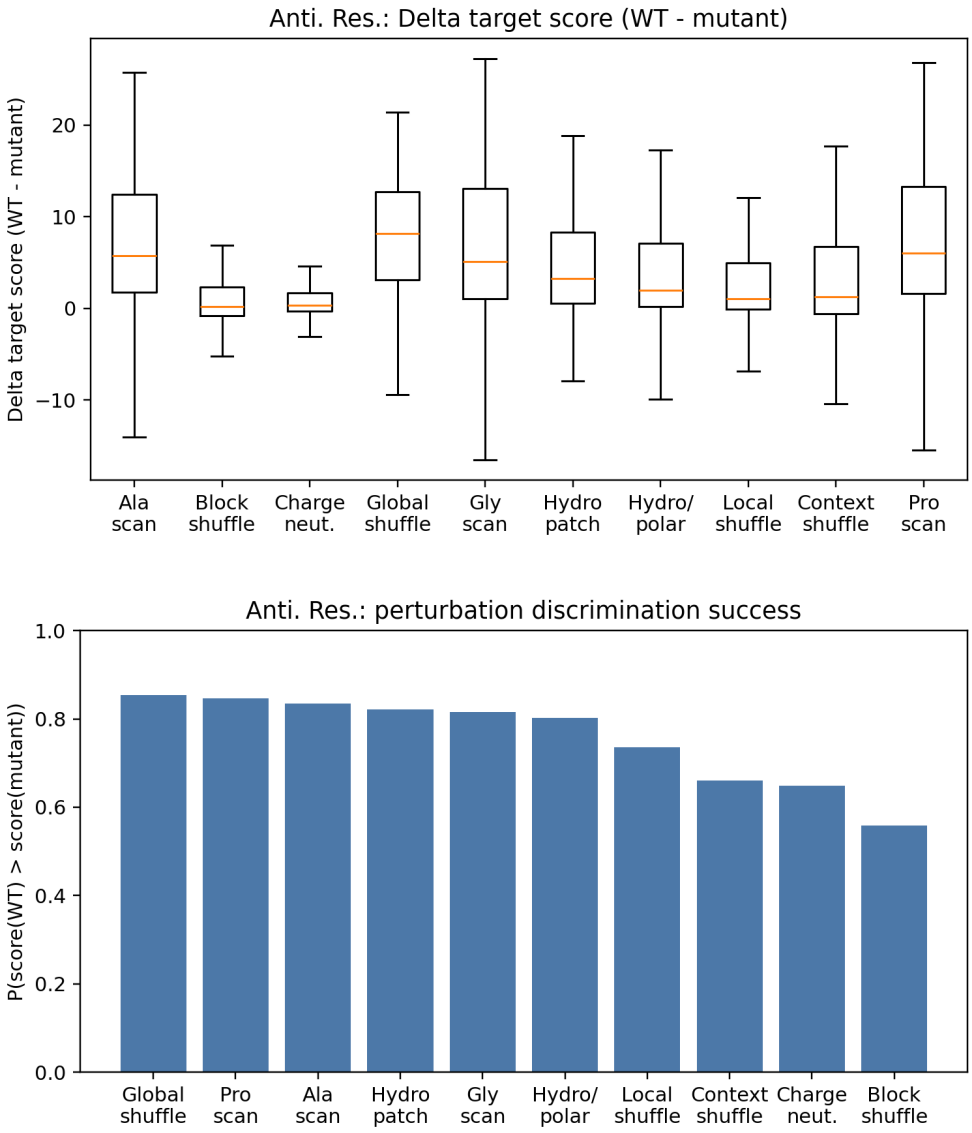


Figure 13: Scalar perturbation diagnostics for Contact and Anti. Res.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Anti. Res.

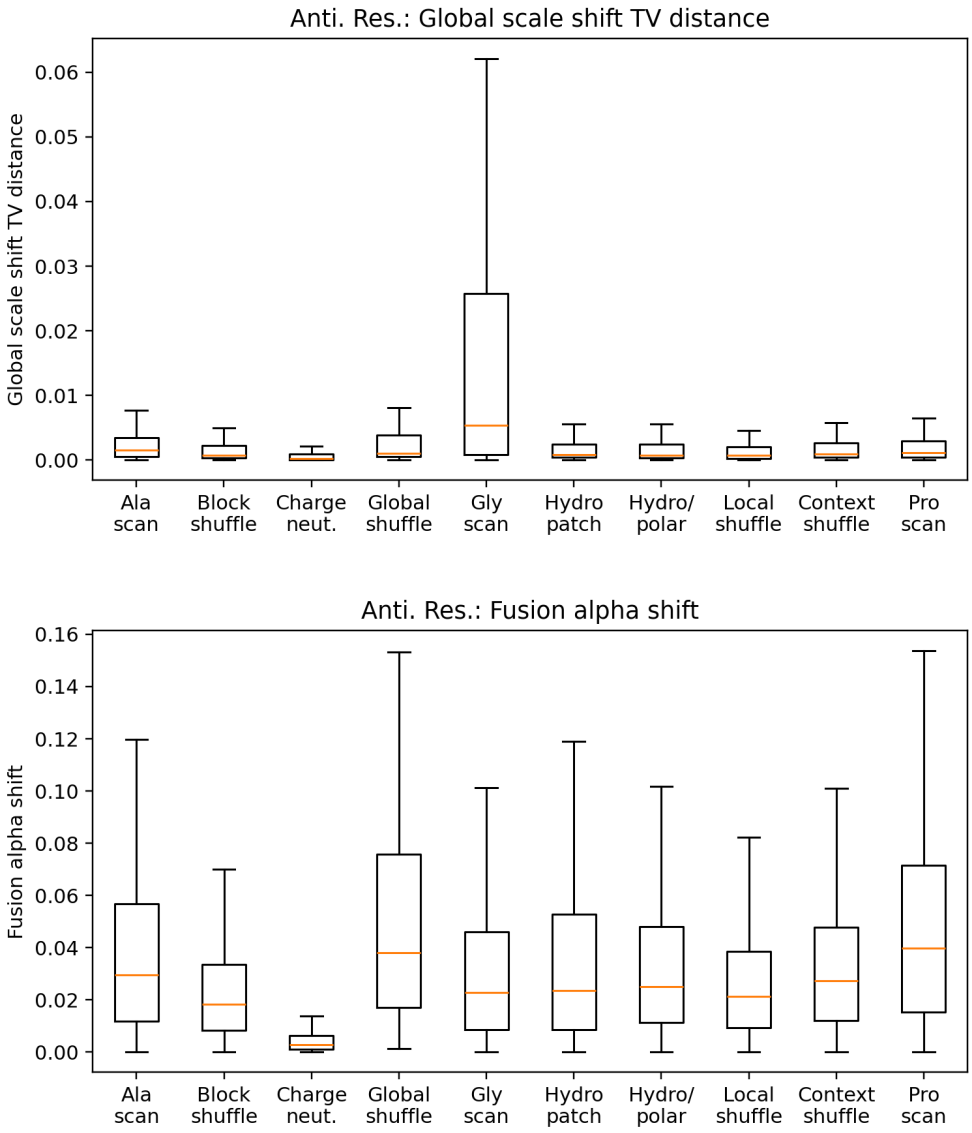


Figure 13: Scalar perturbation diagnostics for Contact and Anti. Res.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

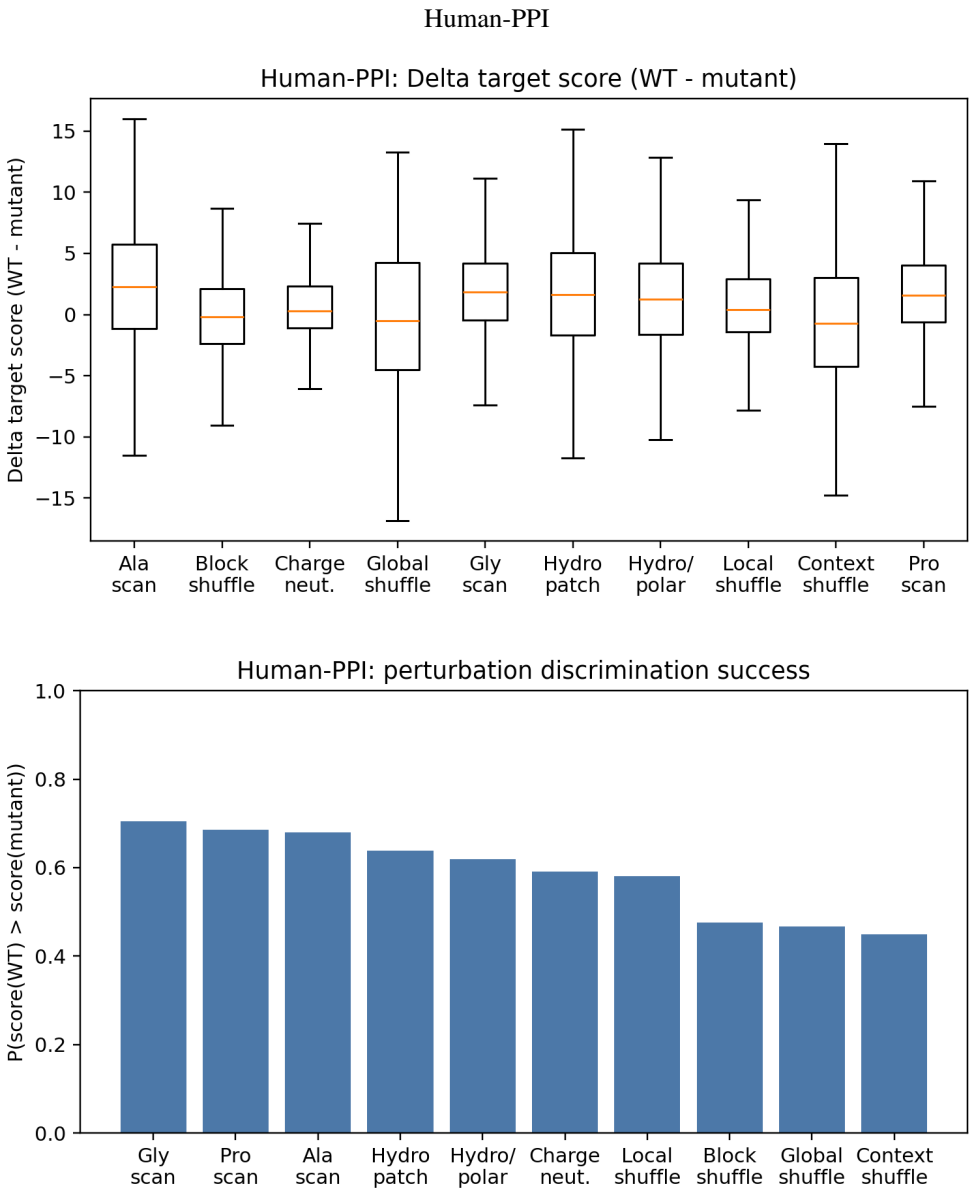


Figure 14: Scalar perturbation diagnostics for Human-PPI and Thermo.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

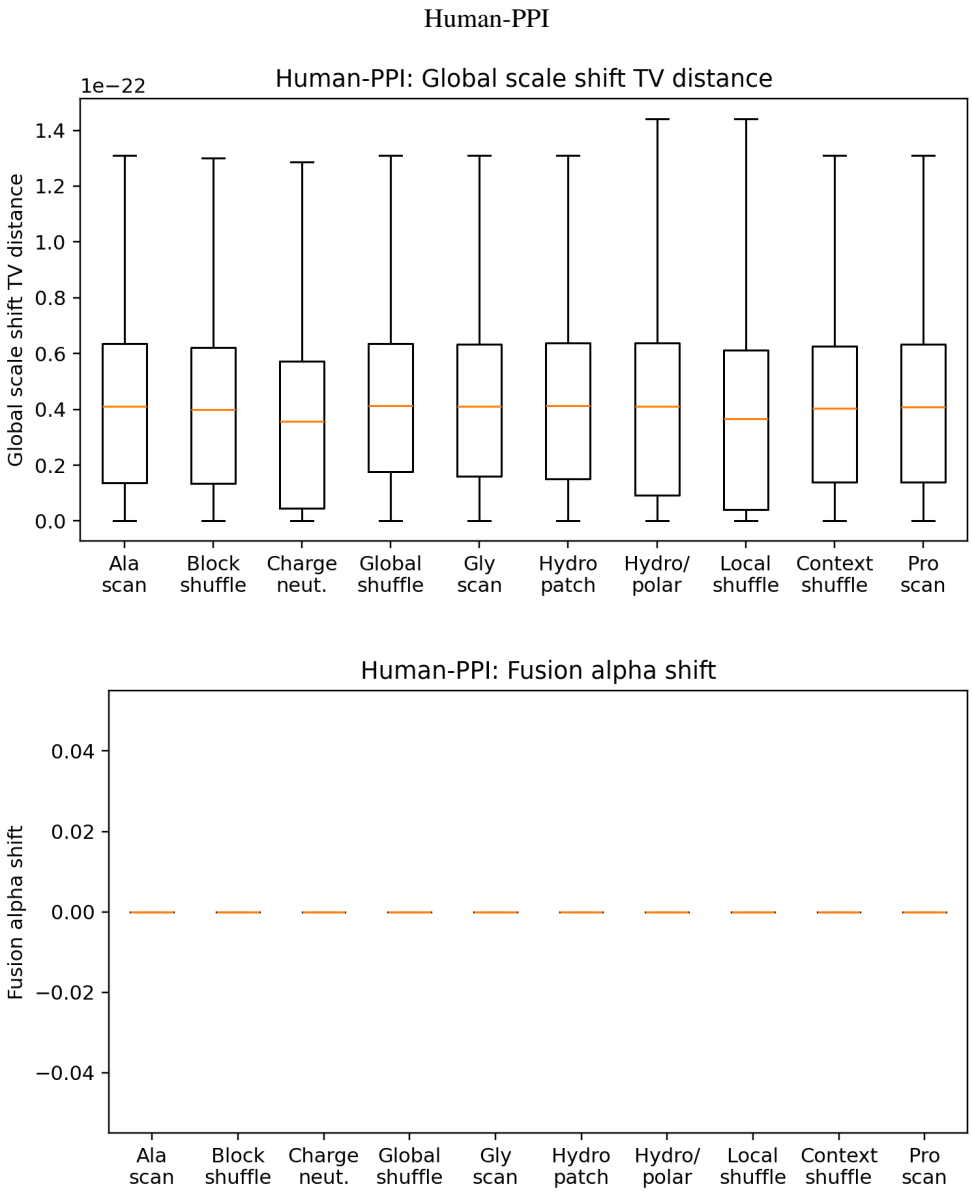


Figure 14: Scalar perturbation diagnostics for Human-PPI and Thermo.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Thermo.

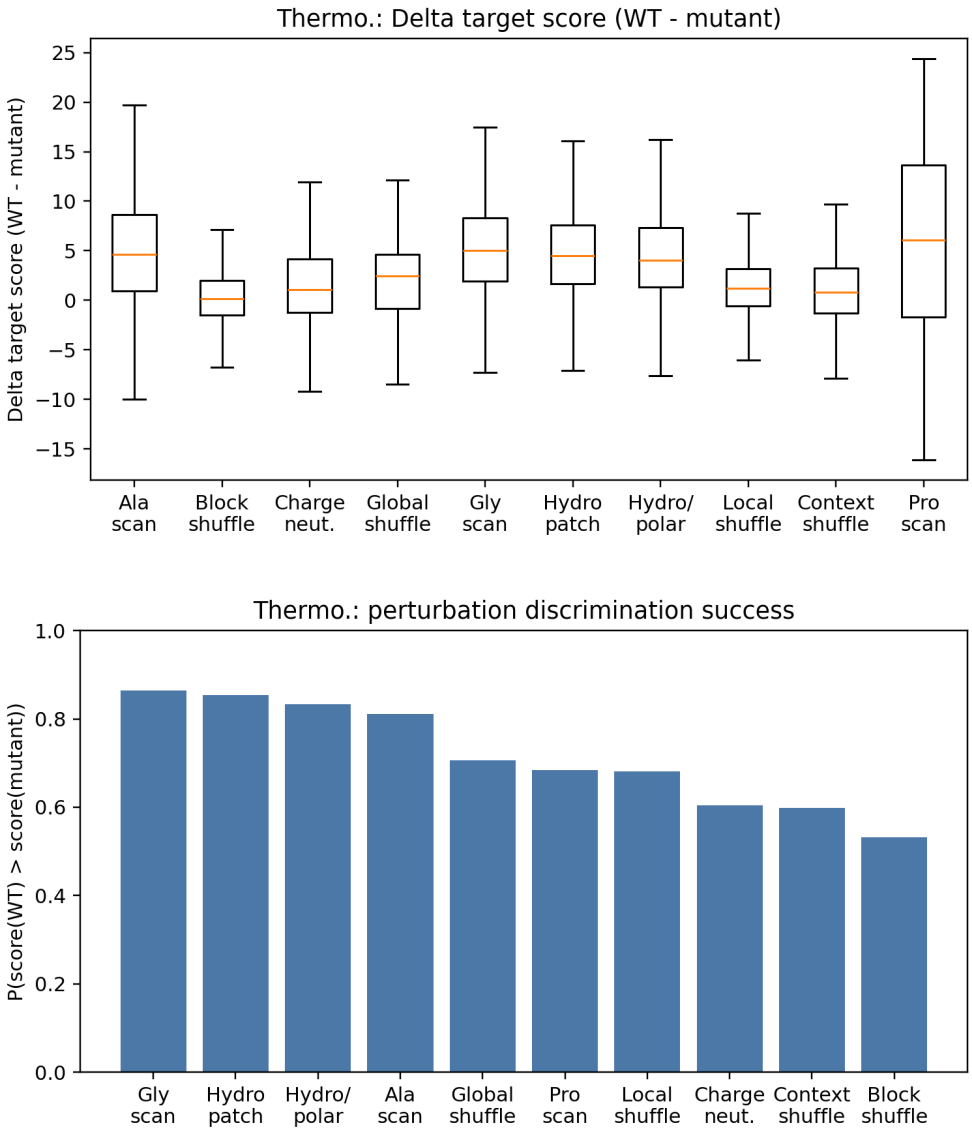


Figure 14: Scalar perturbation diagnostics for Human-PPI and Thermo.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Thermo.

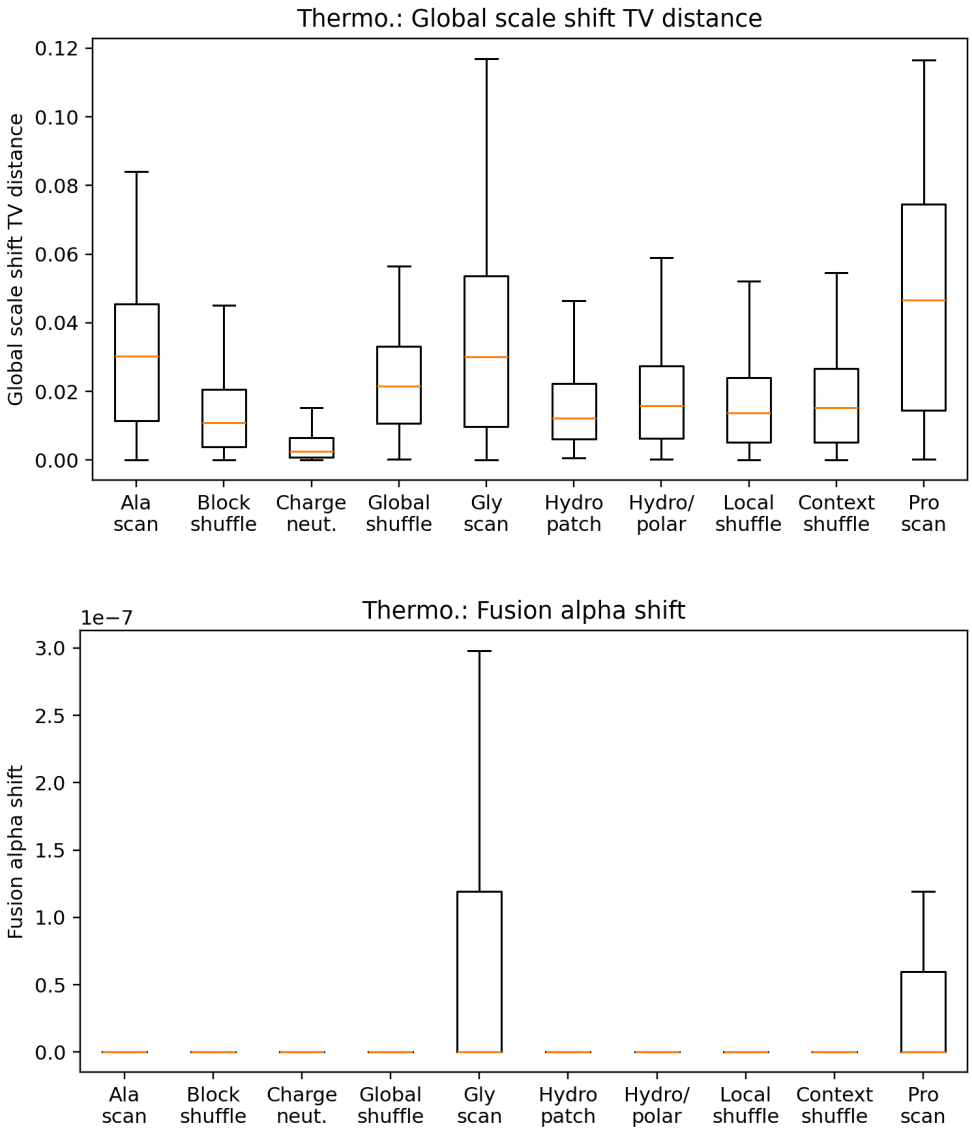


Figure 14: Scalar perturbation diagnostics for Human-PPI and Thermo.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Sor. Sig.

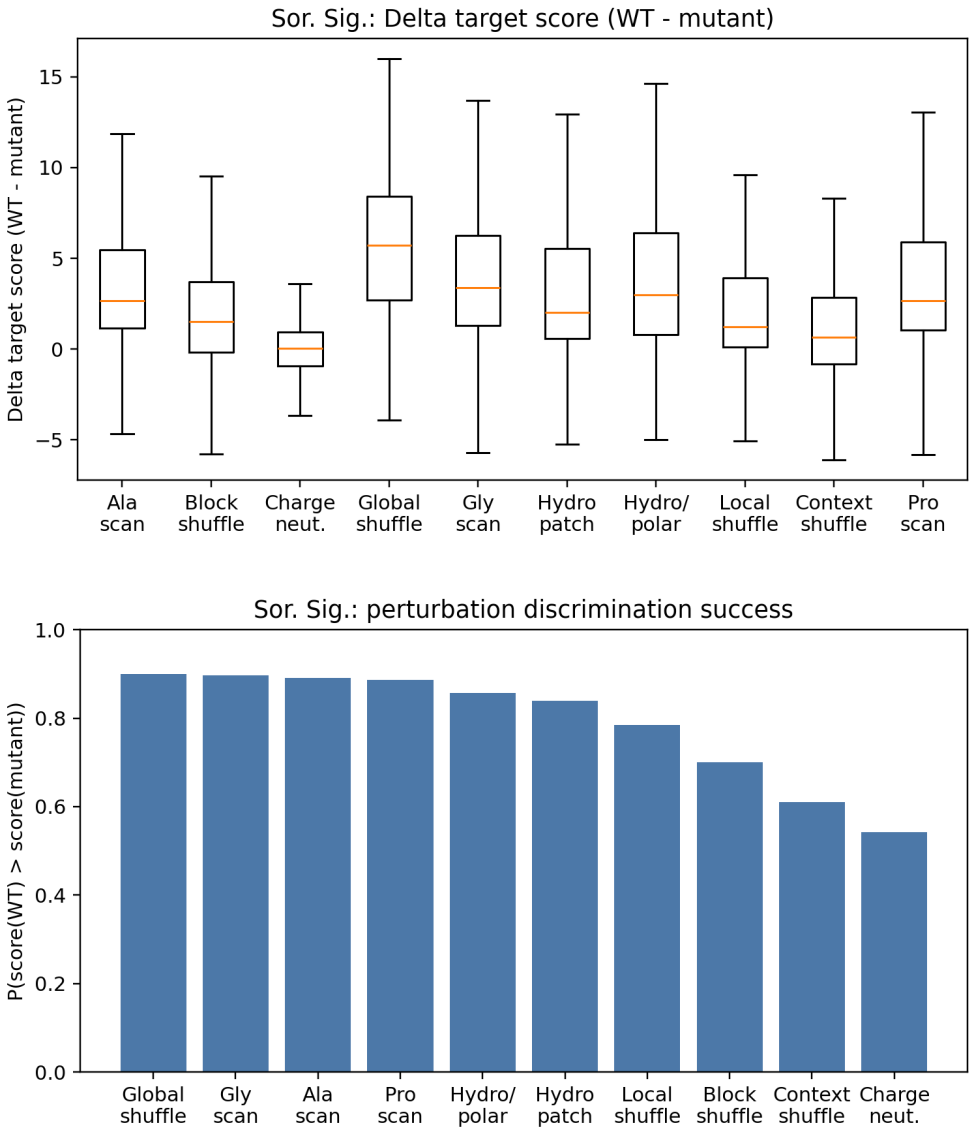


Figure 15: Scalar perturbation diagnostics for Sor. Sig. and M. I. Bin.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

Sor. Sig.

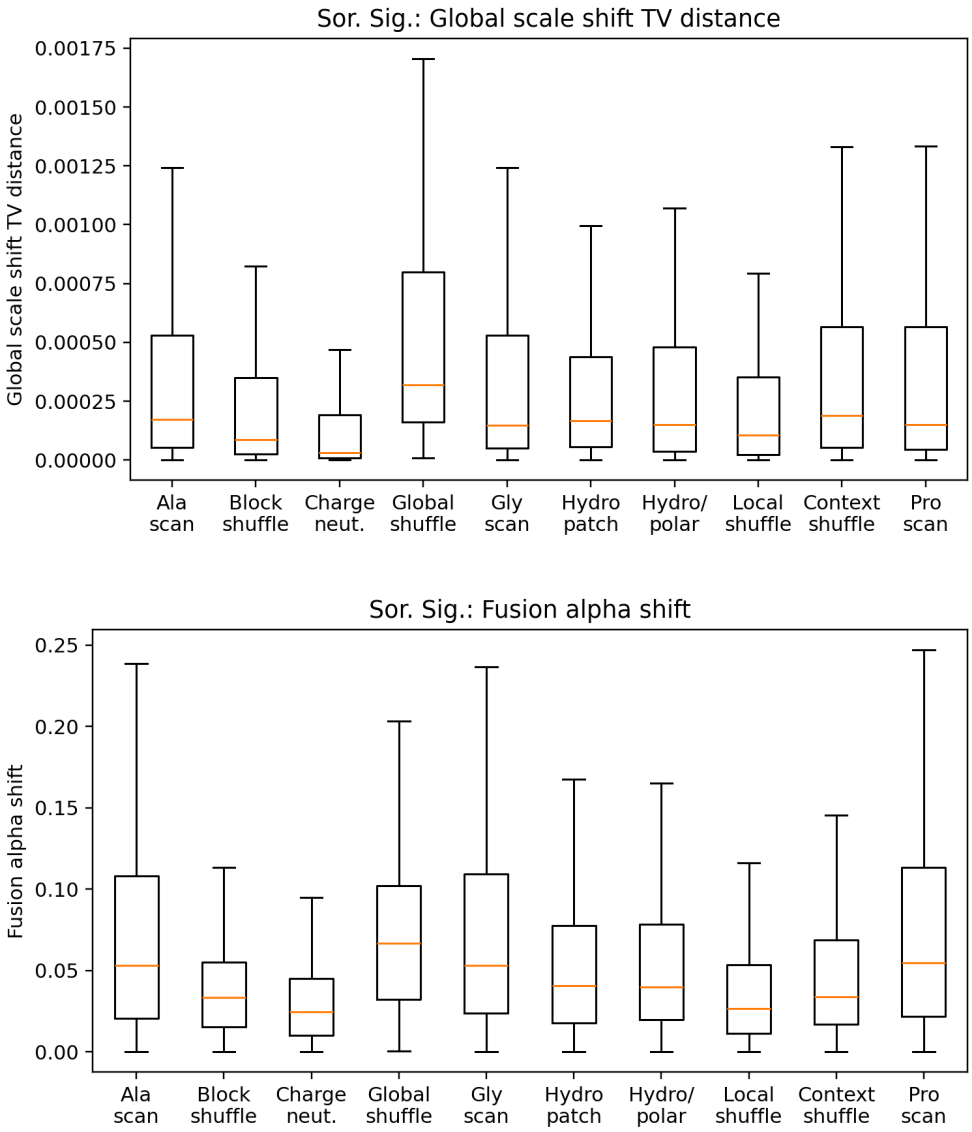


Figure 15: Scalar perturbation diagnostics for Sor. Sig. and M. I. Bin.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

M. I. Bin.

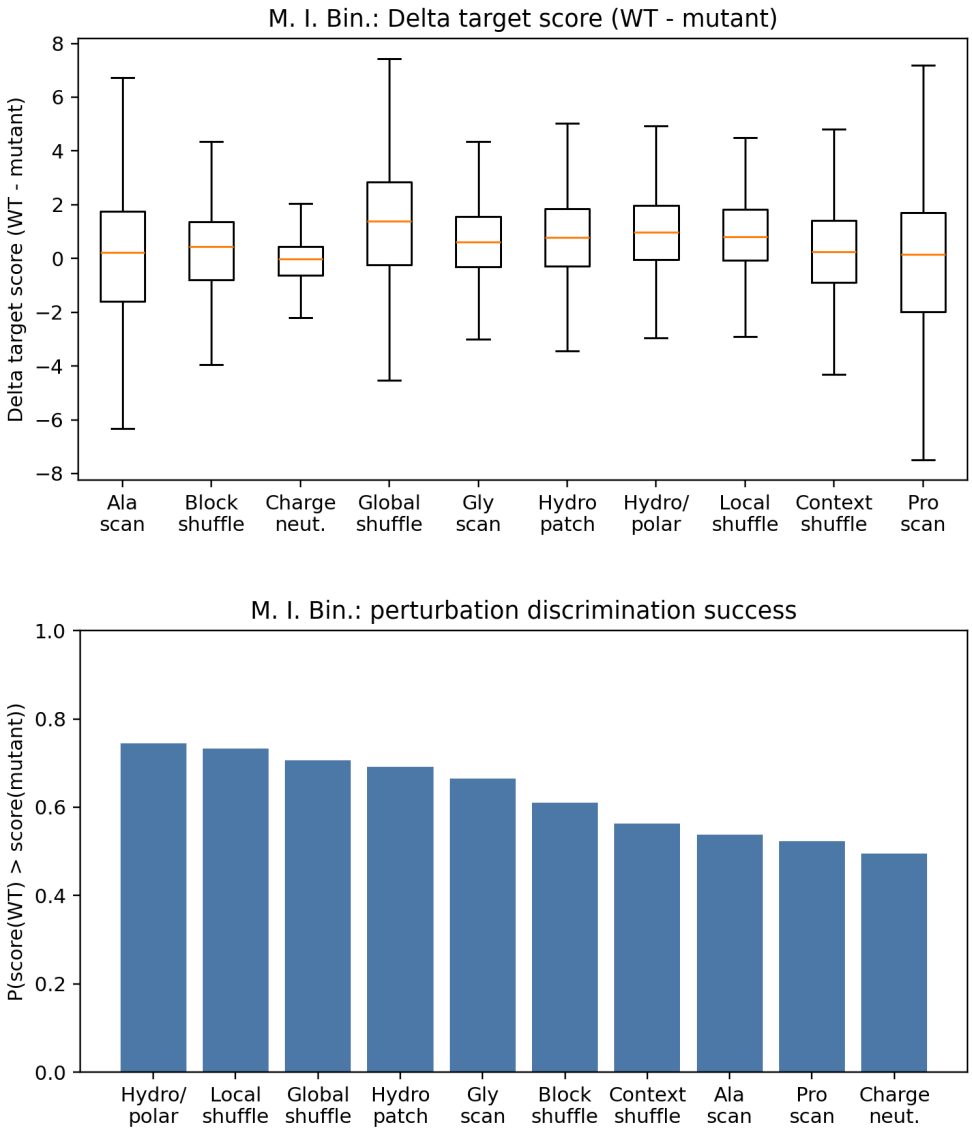


Figure 15: Scalar perturbation diagnostics for Sor. Sig. and M. I. Bin.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

M. I. Bin.

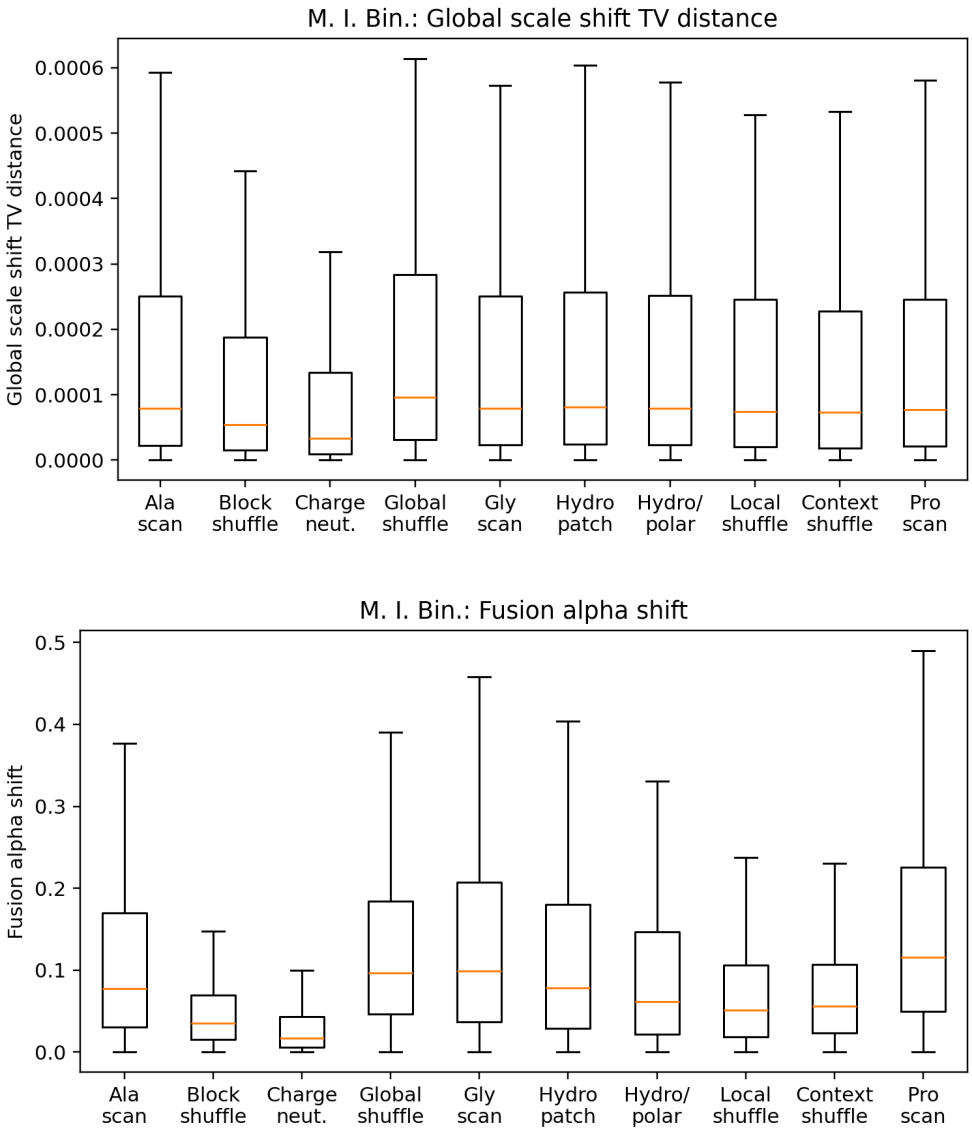


Figure 15: Scalar perturbation diagnostics for Sor. Sig. and M. I. Bin.. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

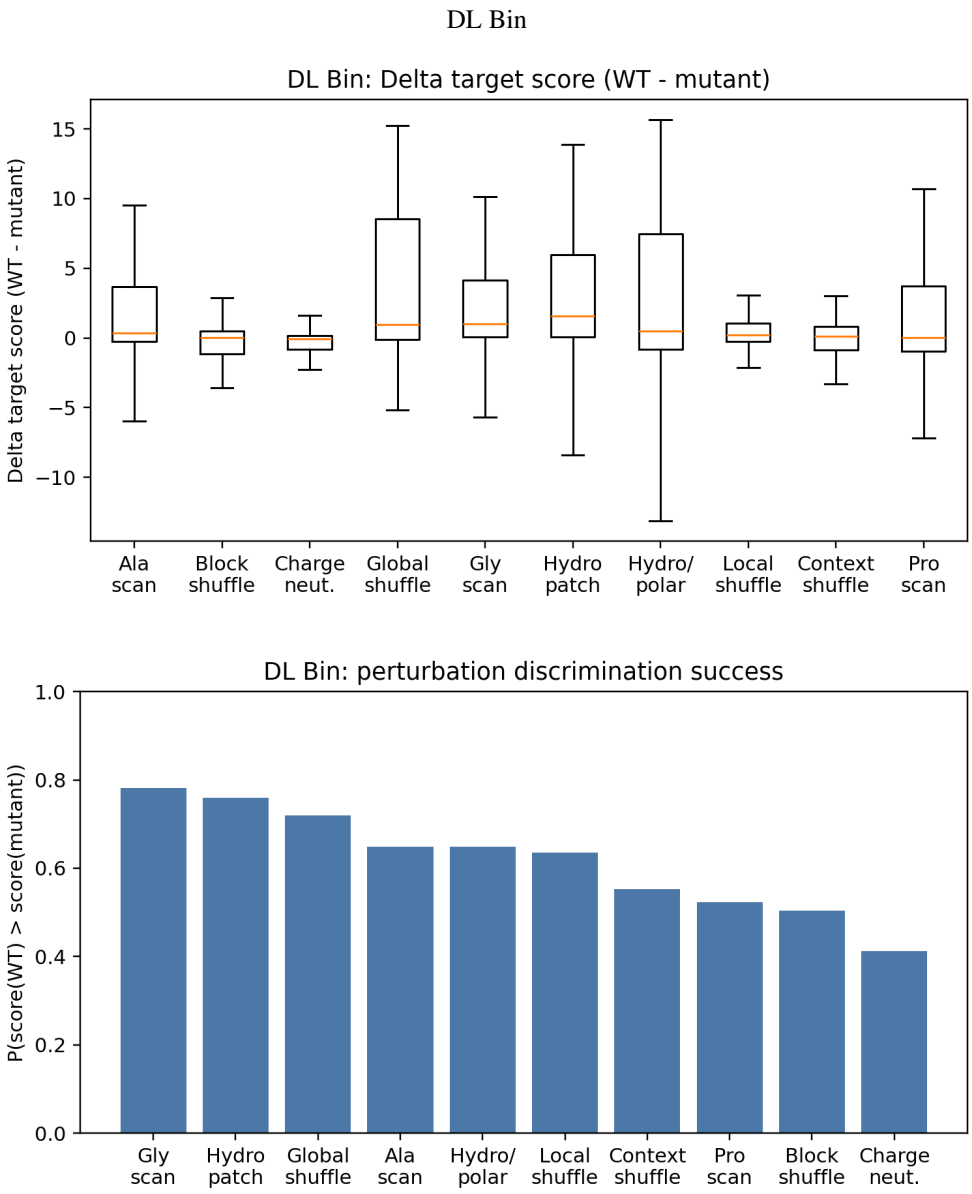


Figure 16: Scalar perturbation diagnostics for DL Bin and DL2 Multi. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

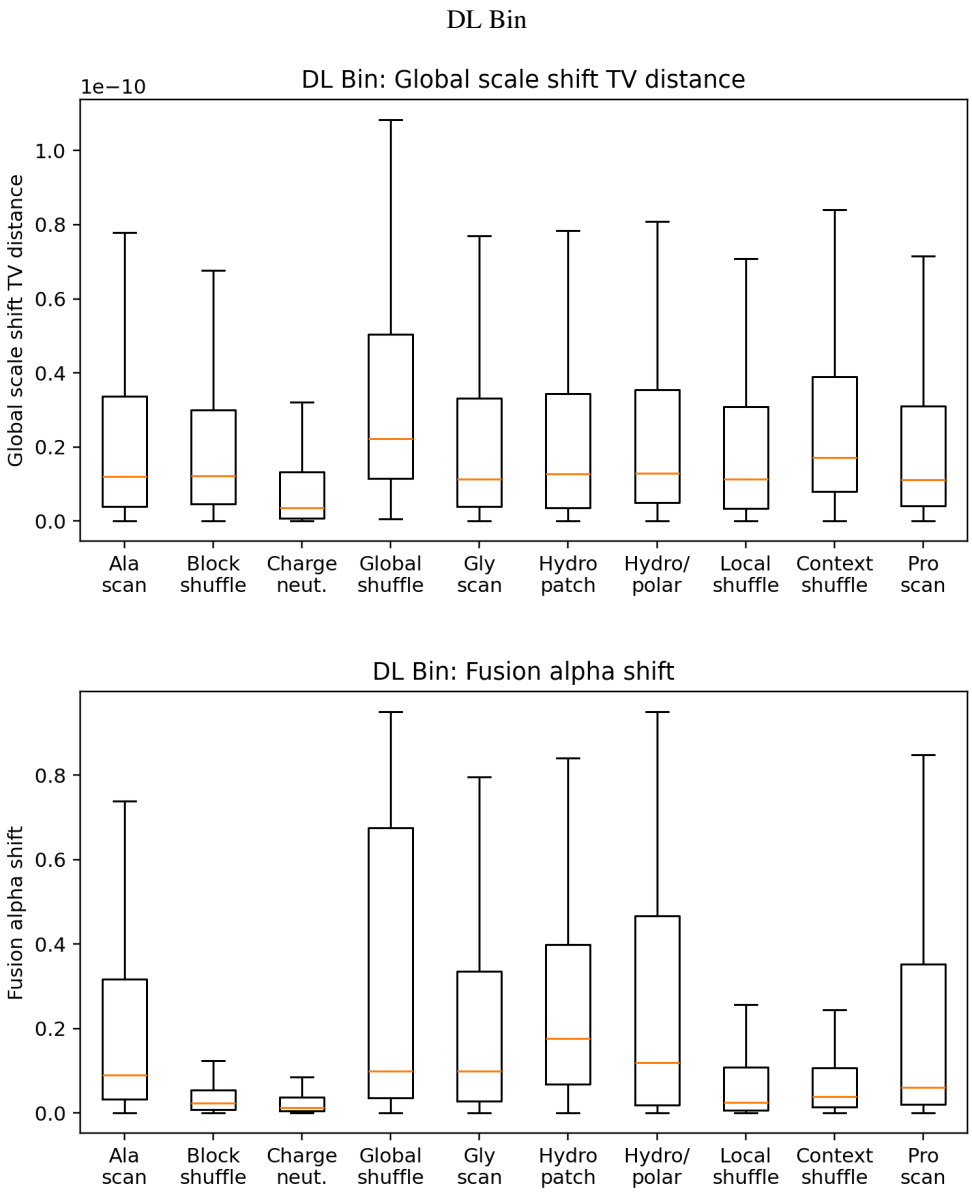


Figure 16: Scalar perturbation diagnostics for DL Bin and DL2 Multi. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

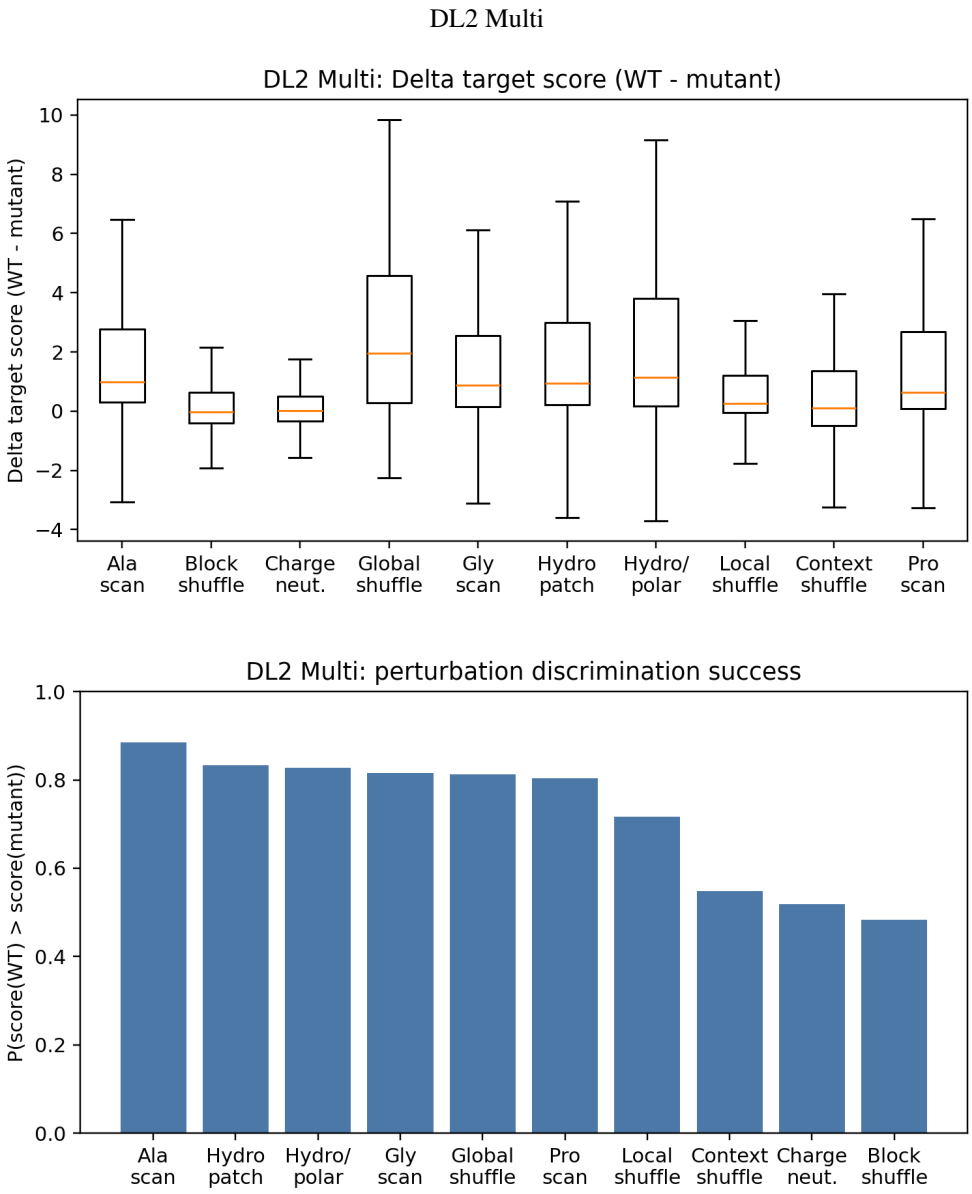


Figure 16: Scalar perturbation diagnostics for DL Bin and DL2 Multi. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

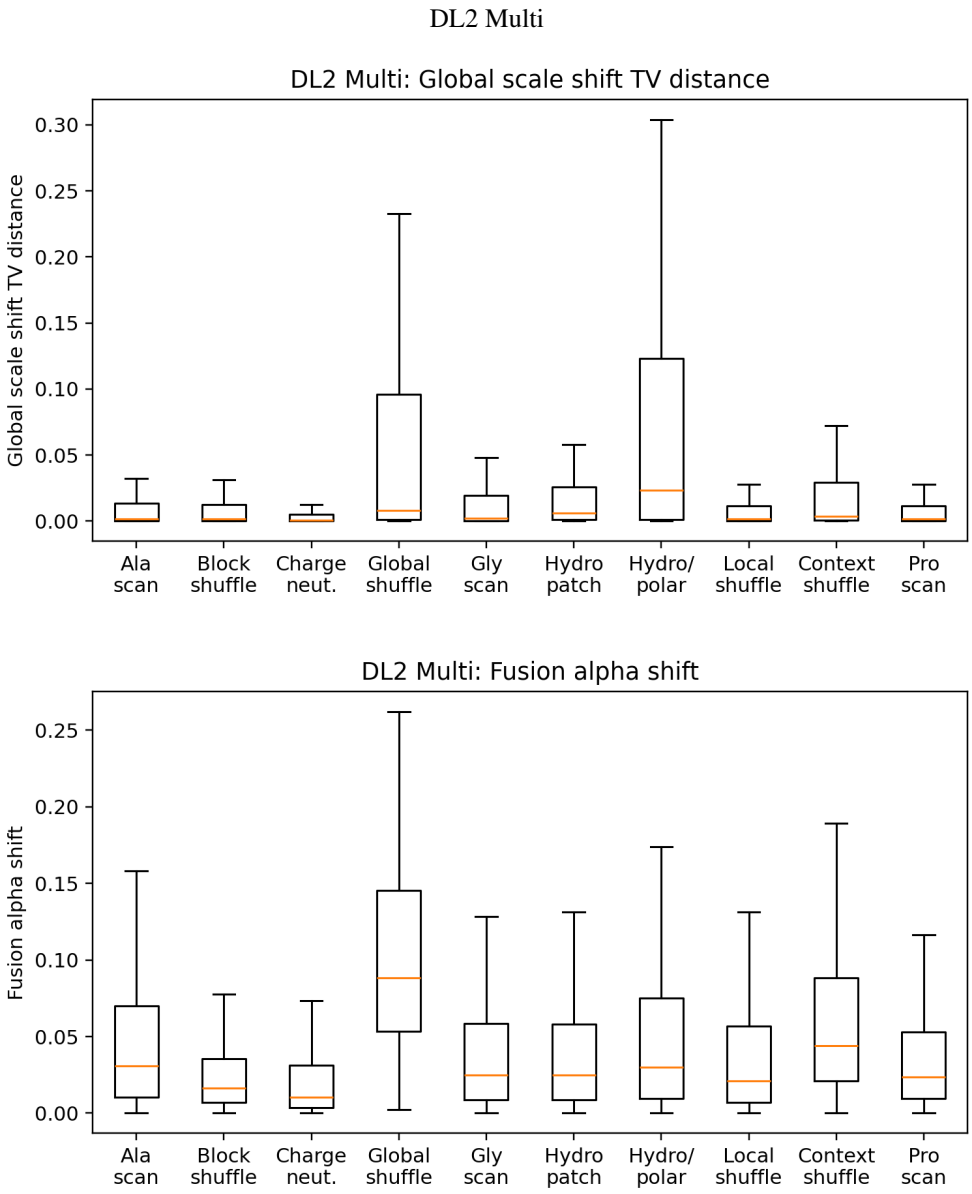


Figure 16: Scalar perturbation diagnostics for DL Bin and DL2 Multi. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

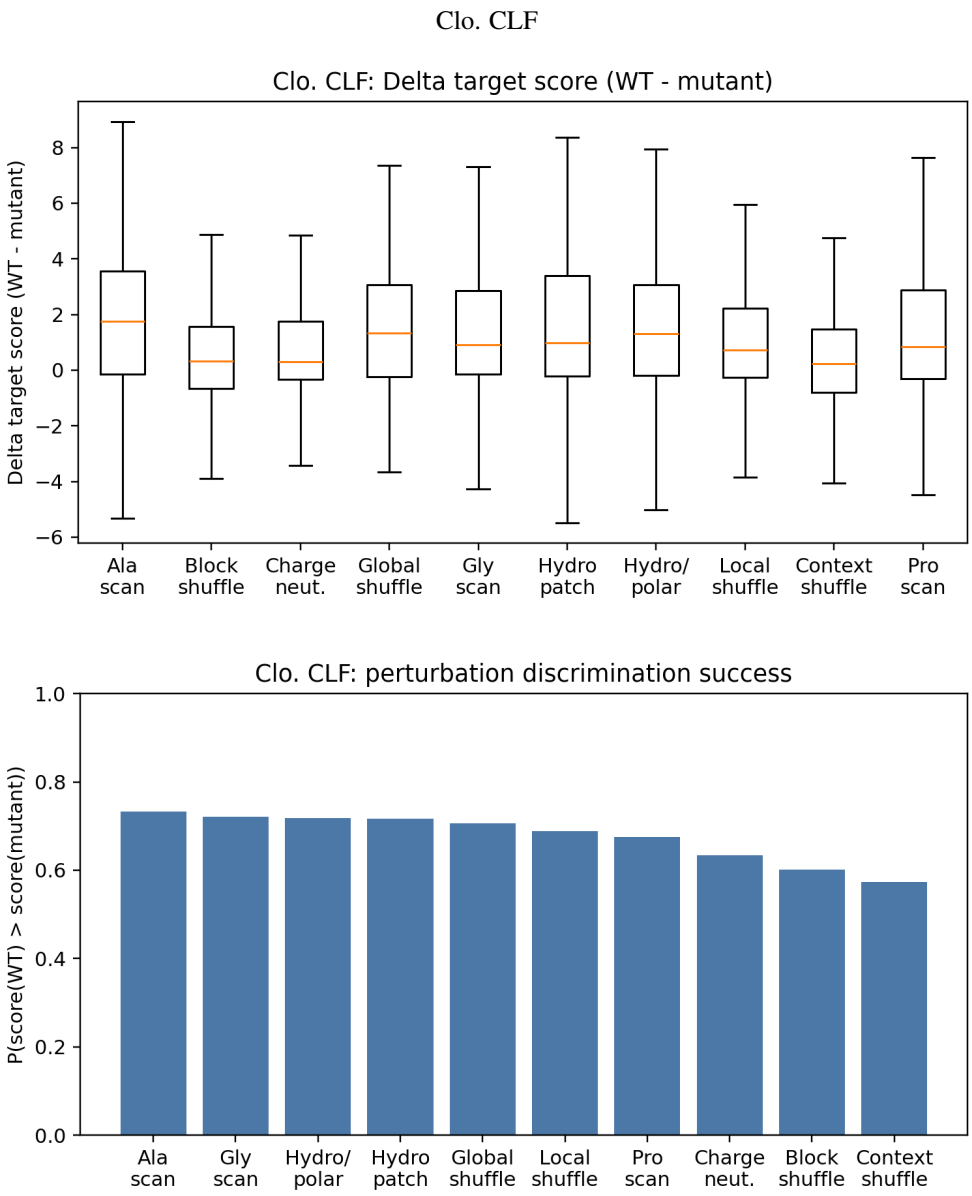


Figure 17: Scalar perturbation diagnostics for Clo. CLF and DeepSoluE. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

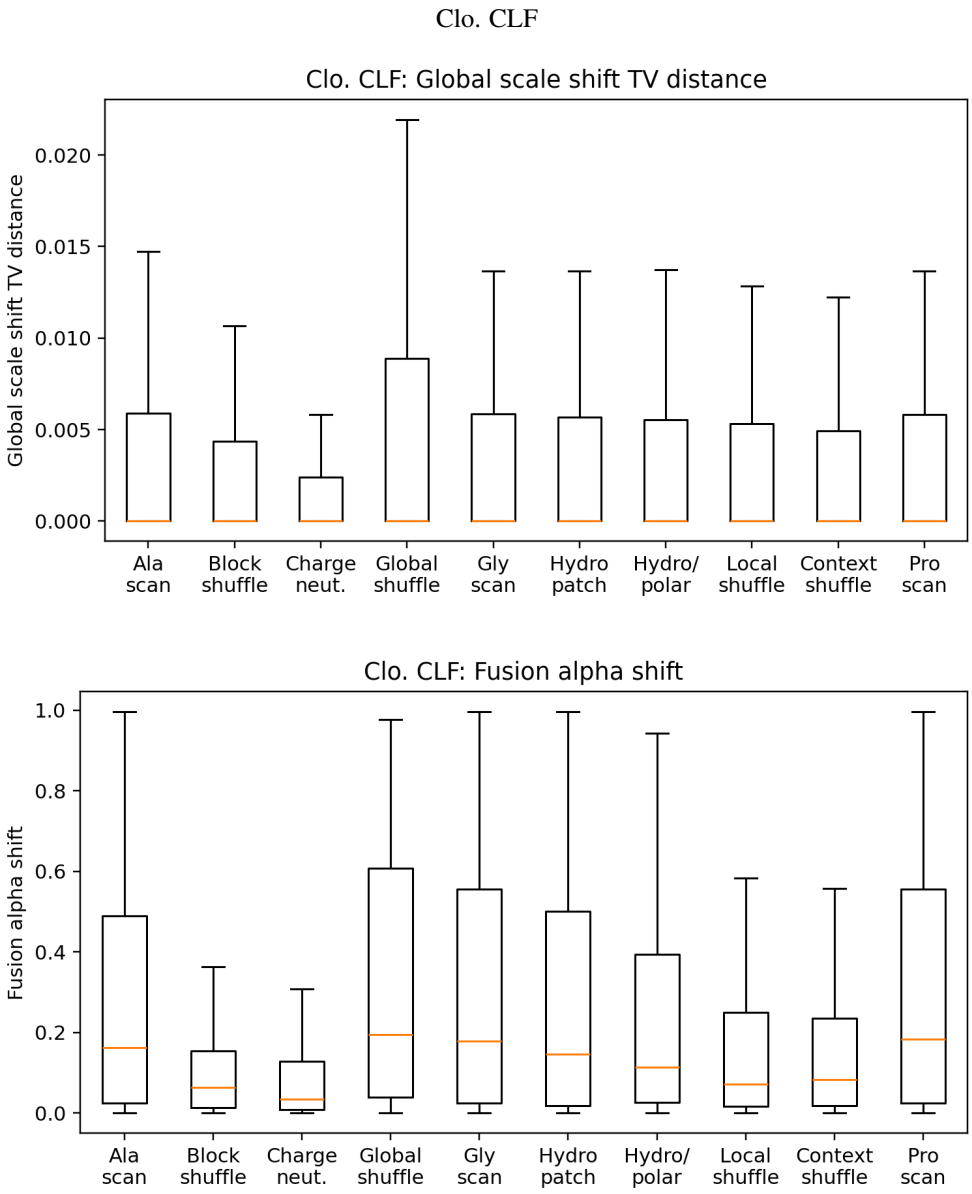


Figure 17: Scalar perturbation diagnostics for Clo. CLF and DeepSoluE. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

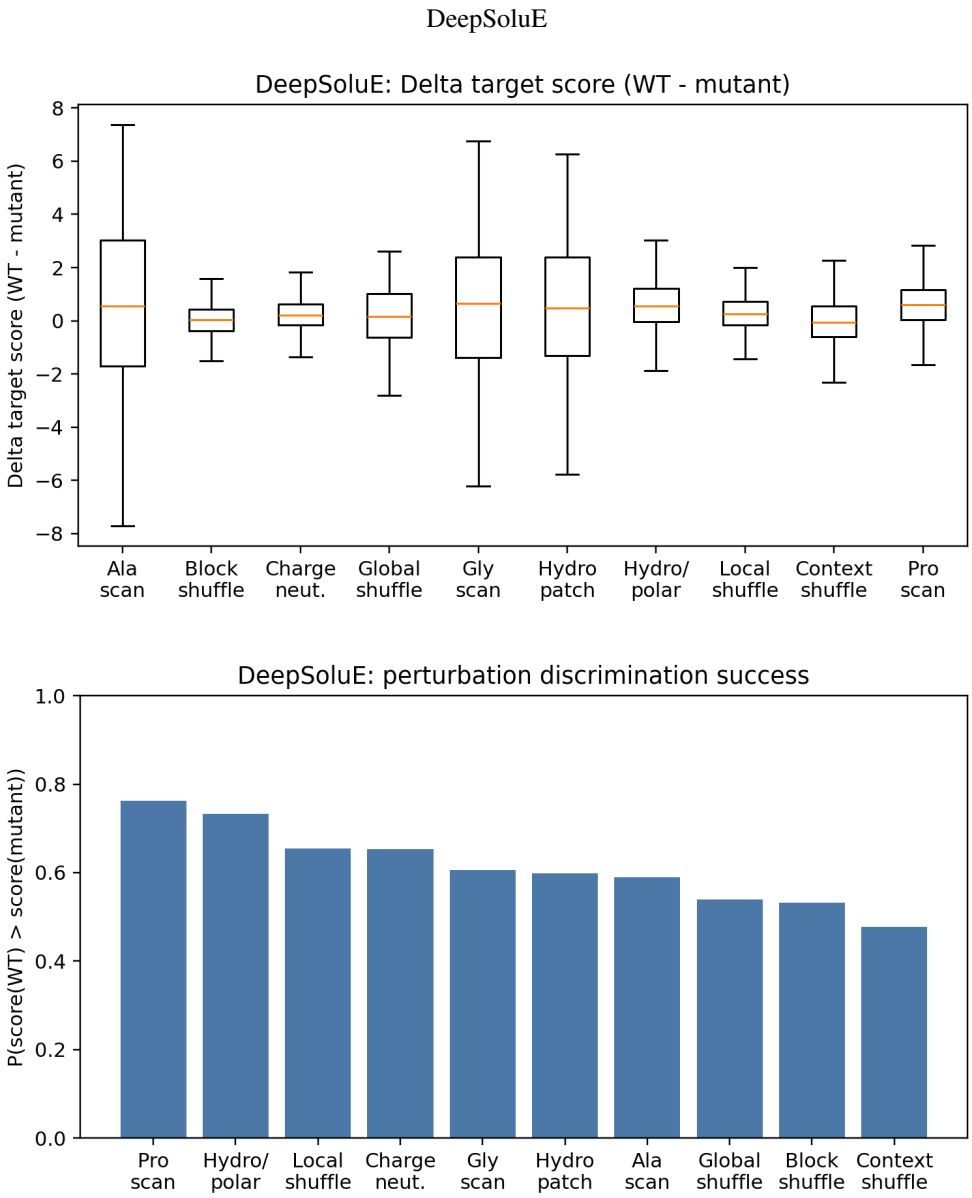


Figure 17: Scalar perturbation diagnostics for Clo. CLF and DeepSoluE. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

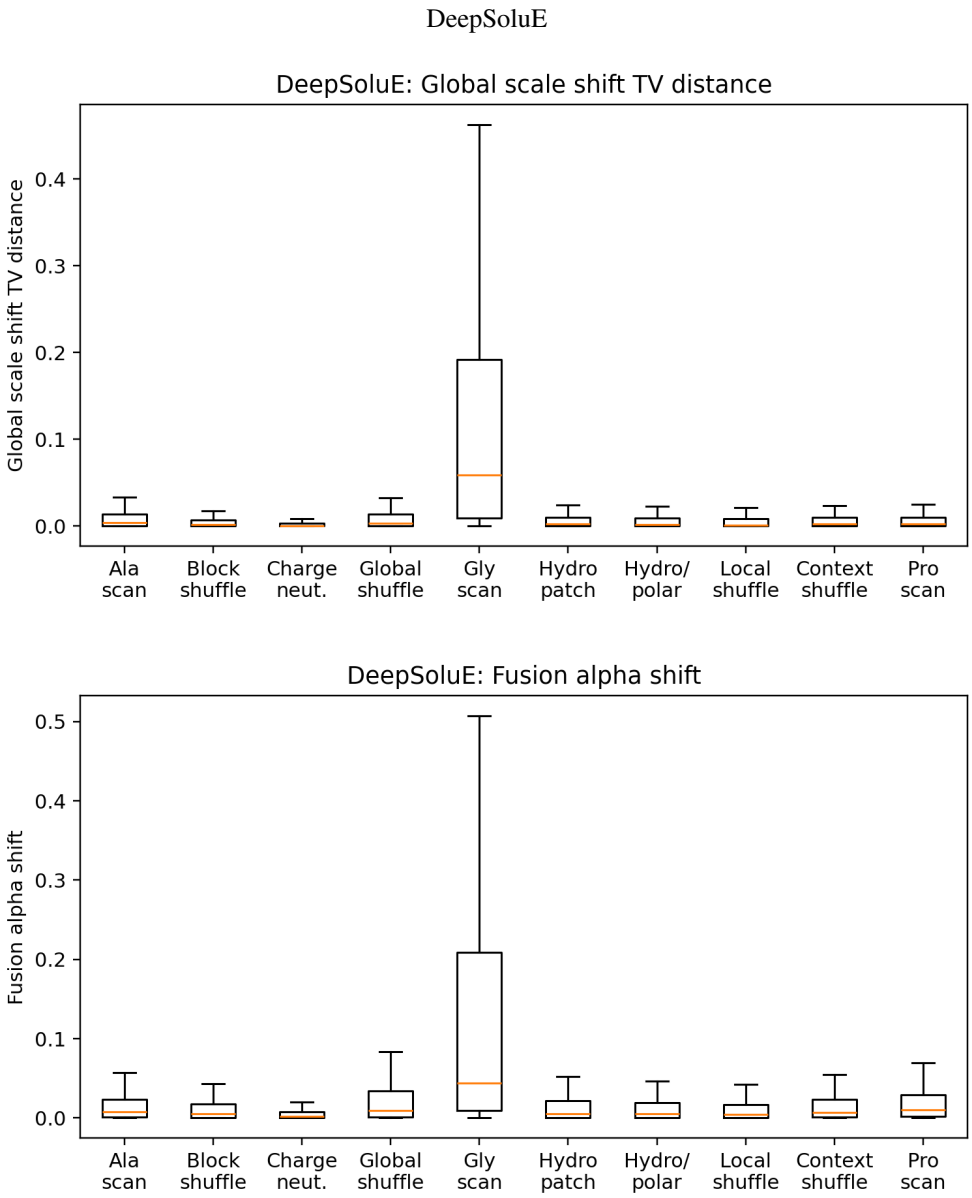


Figure 17: Scalar perturbation diagnostics for Clo. CLF and DeepSoluE. For each task, panels show target-score decrease, success rate, scale-shift TV, and fusion-alpha shift.

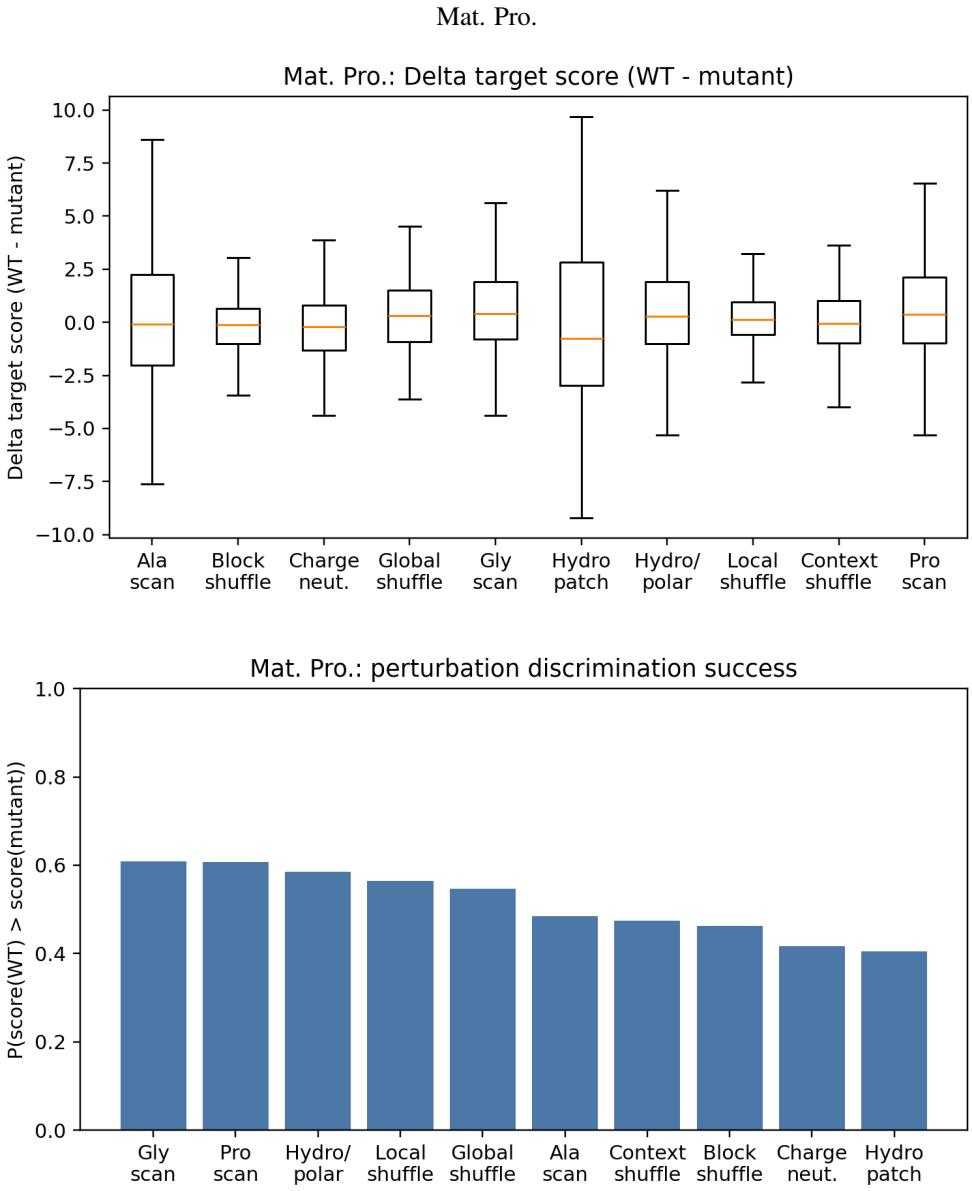


Figure 18: Mat. Pro. task-wise scalar perturbation diagnostics. The four panels show, from left to right and top to bottom, target-score decrease, perturbation success rate, scale-shift total variation distance, and fusion-alpha shift.

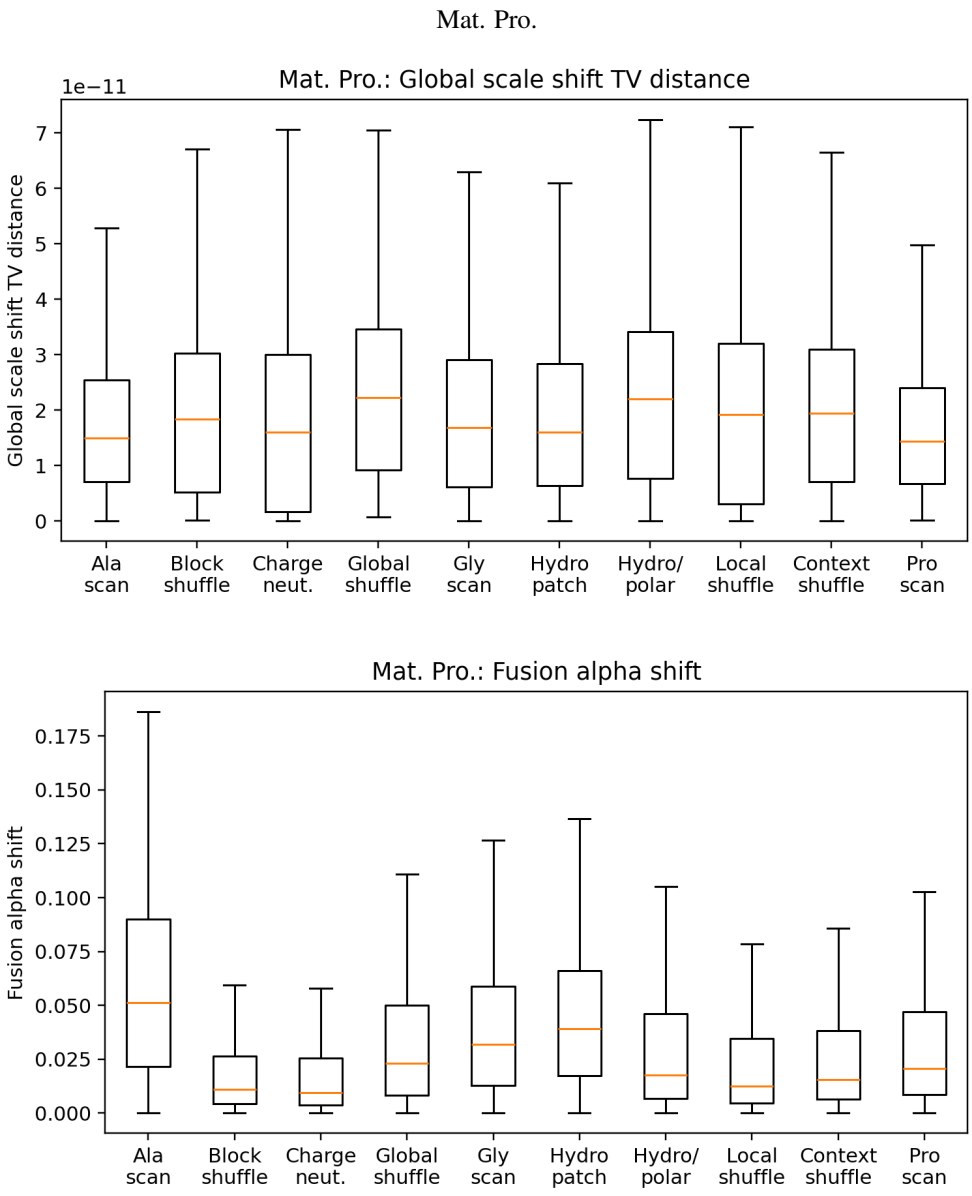


Figure 18: Mat. Pro. task-wise scalar perturbation diagnostics. The four panels show, from left to right and top to bottom, target-score decrease, perturbation success rate, scale-shift total variation distance, and fusion-alpha shift.

Table 17: Structural grouping rules used in the TEM-1 post hoc analysis.

Annotation	Group	Definition
Secondary structure	Helix	DSSP H, G, I
Secondary structure	Strand	DSSP E, B
Secondary structure	Loop/coil	All other DSSP states
Solvent accessibility	Buried	RSA ≤ 0.20
Solvent accessibility	Intermediate	$0.20 <$ RSA ≤ 0.50
Solvent accessibility	Exposed	RSA > 0.50

C.9 Perturbation-site visualization across tasks

Figure 11 shows representative perturbation-site visualizations across all evaluated tasks. Each panel corresponds to one task and one representative test sequence. The x-axis denotes residue position, rows denote perturbation types, and black dots mark mutated residues. This figure complements the Contact example in the main text by showing that the perturbation operators are applied consistently across tasks.

D Structural Context of Adaptive Scale Selection in TEM-1

Appendix C examines the stability of adaptive selection under sequence perturbations; here we ask whether the selected scales are interpretable in a known structural context. The TEM-1 case study links residue-wise scale routing to secondary structure and solvent accessibility, providing a focused view of how MCRA adjusts contextual range within one protein. Representation depth is selected at the task level by the layer-fusion module, whereas this analysis isolates the complementary, residue-dependent scale-selection mechanism.

The analysis uses the TEM-1 sequence (UniProt P62593) and the experimentally resolved mature TEM-1 structure (PDB 1BTL, chain A). Structural annotations are used only for interpretation after model inference; they are not provided as inputs to MCRA during training or prediction.

Sequence-to-structure mapping. MCRA produces residue-level routing probabilities on the model input sequence. The first structural model and chain A are parsed with Biopython 1.85. To compare routing with structural context, the DSSP-derived chain sequence is globally aligned to the full TEM-1 input sequence using `Bio.Align.PairwiseAligner`, with match score 2, mismatch score -1, gap-open score -5, and gap-extension score -0.5. Aligned positions are expressed in the one-based coordinates of the model input sequence and provide the common index for joining routing probabilities, secondary structure, and solvent accessibility. Residues without a matched structure coordinate are excluded from structure-conditioned summaries. This mainly affects sequence regions, such as the signal peptide, that are absent from the mature crystallographic structure.

Secondary-structure grouping. Secondary-structure states are assigned from the PDB structure using the `mkdssp` executable through `Bio.PDB.DSSP`. To obtain a compact three-state annotation, DSSP states H, G, and I are grouped as α-helix; E and B are grouped as β-strand; all remaining states, including turns, bends, and irregular coil states, are grouped as loop/coil. These three groups are used in the secondary-structure panels of the TEM-1 analysis.

Solvent-accessibility grouping. Relative solvent accessibility (RSA) is read from DSSP, which normalizes the observed solvent-accessible surface area by a residue-specific maximum accessible surface area. We use three coarse groups: residues with RSA ≤ 0.20 are treated as buried, residues with $0.20 <$ RSA ≤ 0.50 as intermediate, and residues with RSA > 0.50 as exposed. These thresholds provide a simple structural stratification for testing whether routing preferences differ between protein core and surface environments.

Routing-scale aggregation. Each MCRA context expert is assigned to one of four receptive-field scale groups according to its convolutional receptive-field kernels: Small $(3, 5, 9)$, Medium $(3, 7, 15)$, Large $(5, 9, 17)$, and XLarge $(7, 11, 15)$. For residue i, the probability of scale group s is obtained by aggregating the router probabilities of all experts in that group,

$$P_i(s) = \sum_{e \in \mathcal{E}_s} p_i(e), \tag{30}$$

where $p_i(e)$ is the routing probability assigned to expert e at residue i and $\mathcal{E}_s$ denotes the expert set of scale group s. The dominant scale is $\arg\max_s P_i(s)$, and routing confidence is $\max_s P_i(s)$. For a structural group such as buried residues or α-helices, scale preference is the mean of $P_i(s)$ over all residues belonging to that group.

Sequence-level routing pattern. Figure 10 (left) shows that Large-scale routing forms the dominant background across most of TEM-1, rather than being confined to a single motif or secondary-structure class. This preference is not uniform: localized Small- and Medium-scale patches interrupt the Large-scale profile, and the later part of the sequence exhibits a more heterogeneous scale mixture. XLarge routing remains negligible at most positions. The model therefore uses a broad-context default while retaining residue-level adjustments, rather than assigning one fixed scale to the entire protein.

Association with secondary structure. The middle panel quantifies this pattern after alignment to DSSP annotations. Large remains dominant in α-helices (0.72), β-strands (0.74), and loop/coil regions (0.85). However, ordered secondary structures retain more Medium-scale routing than loops: 0.22 for helices and 0.17 for strands, compared with 0.07 for loops. Small-scale probabilities remain modest $(0.06$–$0.09)$, and XLarge is effectively absent. Thus, the dominant scale is shared across structural classes, whereas the secondary contribution changes with local structural organization.

Association with solvent exposure. The right panel reveals a clearer accessibility-dependent shift. Buried residues allocate 0.68 probability to Large and 0.23 to Medium, whereas intermediate and exposed residues increase Large routing to 0.84 and 0.88 and reduce Medium routing to 0.09 and 0.08, respectively. Small routing also decreases from 0.08 in buried residues to 0.05 in exposed residues. This monotonic redistribution indicates that the model preserves more intermediate-context processing in the protein interior while favoring broader sequence context at solvent-exposed positions.

Together, the three views provide a consistent account: TEM-1 has a global Large-scale preference, but its routing distribution is locally adjusted and systematically associated with structural environment. Because MCRA receives only the amino acid sequence, these associations suggest that its adaptive routing recovers sequence cues related to structural context. They do not, however, establish a causal structural mechanism or general behavior beyond this single-protein case study.

Interpretation scope. The TEM-1 analysis is intended as a qualitative mechanistic probe. It evaluates whether learned routing patterns align with known structural environments after the Anti. Res. model predicts the correct *Betalactams* class for P62593. Because the structural annotations are used only after inference, the observed association between routing scales, secondary structure, and solvent exposure should be interpreted as post hoc evidence of an association between adaptive scale selection and structural context, rather than as supervised structural training signal. This single-protein analysis does not control for amino-acid identity or correlation among neighboring residues. A population-level structural claim would require additional proteins, controlled association tests, and router baselines; the present case is illustrative.

E Interpretability and Stability of Adaptive Scale and Depth Selection

The task-level analyses in this appendix complete the interpretability and stability account developed in Appendices C and D. Expert-group usage measures which receptive-field scales are selected across tasks, and layer-fusion weights measure which pretrained representation depths are emphasized. The representation-space shuffled controls then test whether the resulting fused features organize real protein sequences differently from composition-matched sequence permutations. These

views jointly characterize adaptive scale and depth selection without treating a two-dimensional projection as evidence of predictive performance.

E.1 Task-level expert and scale preferences

Task-level scale-preference figures are computed from held-out test sequences using the best-validation checkpoint. For task t, let N_t be the number of evaluated proteins, L_n the number of valid residues in protein n, and $r_{n,i,e}$ the router probability assigned to expert e. We first average within each sequence and then across sequences,

$$u_{t,e} = \frac{1}{N_t} \sum_{n=1}^{N_t} \left(\frac{1}{L_n} \sum_{i=1}^{L_n} r_{n,i,e} \right). \tag{31}$$

This gives equal weight to proteins rather than allowing long proteins to dominate the task summary. Expert values are normalized to sum to one within a task. Scale usage is then

$$U_{t,g} = \sum_{e \in \mathcal{E}_g} u_{t,e}, \tag{32}$$

using the four expert groups defined in Appendix C.5. The preferred task scale is $\arg\max_g U_{t,g}$. These labels describe the relative use of learned receptive-field groups; they do not assign a unique physical length or a known structural class to a task. The argmax is only a summary of the largest component and does not imply that a task selects a single scale. Several tasks retain substantial probability across multiple scale groups, and routing can also vary among residues within the same protein. Consequently, choosing one fixed profile on the validation set cannot generally reproduce MCRA's residue-wise multi-scale mixture; the fixed-scale ablation instead evaluates a uniform context operator applied to all residues. Equal protein weighting prevents long sequences from dominating the average, but does not remove an association between sequence length and routing. Length-matched or length-stratified analyses and an untrained-router reference are not included. Moreover, four scale totals cannot establish balanced use of all eight experts. The quantities $u_{t,e}$ and residue-level Top-4 selection frequencies should be distinguished when evaluating expert collapse; we do not infer per-expert utilization from the scale-group plots.

E.2 Layer-fusion preferences

The last-16-layer fusion module contains one learned scalar w_k for each selected ESM2 layer. The task-level weight shown in the analysis is

$$a_{t,k} = \frac{\exp(w_{t,k})}{\sum_{j=1}^{16} \exp(w_{t,j})}. \tag{33}$$

Weights are read from the best-validation checkpoint and therefore sum to one for each task. The uniform reference in the figure is $1/16$. Because these weights are task parameters rather than input-dependent router outputs, they are constant across WT and mutant sequences for a fixed checkpoint. Layer preferences should therefore be interpreted as task-level representation-depth selection, complementary to the residue-dependent scale routing in Appendix E.1.

E.3 Representation-space and shuffled-control analysis

The representation-space analysis compares four features extracted from the corresponding trained models: the fused ESM2–MCRA representation consumed by the task head, a fixed-scale single-context representation, the MCRA side-branch representation, and the final-layer ESM2 representation. Padding positions are removed and residue features are pooled with the same valid-token mask used by the downstream model.

For each real protein, up to five shuffled controls are generated by permuting the amino-acid order while preserving sequence length and exact amino-acid composition. Duplicate shuffles and shuffles identical to the source sequence are rejected. Control generation uses seed 42. For the combined visualization, at most 500 real and 250 shuffled examples are sampled per task with the same seed. All selected task and control embeddings of a common dimensionality are embedded jointly. The t-SNE

projection uses cosine distance, PCA initialization, automatic learning-rate selection, perplexity 30 (reduced automatically for a small input set), and random state 42.

t-SNE preserves local neighborhoods only approximately and can change with sampling and initialization. The panels are therefore qualitative diagnostics of representation organization, not quantitative tests of cluster separation. The task adapters are trained independently, so equal embedding dimensionality does not imply aligned feature coordinates across tasks. In the joint plot, task clusters can therefore reflect separately learned coordinate systems. Real–shuffled separation is also not specific to adaptive routing. Performance claims rest on held-out task scores, not apparent two-dimensional separation.

Figure 8 applies the same projection and shuffled-control protocol to all 13 tasks. Colored points are real proteins, with color indicating the residue neighborhood score (RNS), while gray points are composition-preserving shuffled controls. The task-wise panels reveal whether the learned fused space places biologically ordered sequences in localized neighborhoods distinct from the shuffled background. Differences in geometry across tasks are expected because adaptive depth and scale selection is optimized for each task; comparisons should therefore focus on real-versus-control organization within a panel, not on absolute t-SNE coordinates across panels.

E.4 Fusion-gate dynamics

For the gate β_i defined in Appendix C.5, the task-level mean is computed over valid residues and evaluation examples. The ESM2-dominant fraction is the fraction of valid positions with $\beta_i > 0.5$; the complementary fraction is defined as MCRA-dominant. Curves over epochs use the validation logs produced during training, whereas final task summaries use the selected checkpoint. These quantities describe how the model allocates weight between pretrained and adapted features. They do not by themselves establish that one branch encodes a specific biological mechanism. Their interpretation also depends on the magnitudes of the projected branch features: a larger gate weight need not imply a proportionally larger causal contribution to the prediction. A direct gate-removal comparison is not included in the present component table; the gate curves characterize the fitted mixture rather than an isolated intervention on the gate.

AI Use Statement

Generative AI tools assisted manuscript revision, and consistency checks of reported configurations and tables. They were not used to run training experiments or generate new experimental measurements. The authors are responsible for verifying the cited sources, scientific interpretations, and final content.