PROTGROUND: FUNCTIONAL PHRASE GROUNDING IN LOCAL PROTEIN 3D STRUCTURAL REGIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Protein functions often depend on local 3D environments, making it important to localize functional descriptions to their supporting structural regions. However, existing protein–text pretraining methods mainly operate at the whole-protein level, leaving these local correspondences unresolved. To this end, we introduce ProtGround, a fine-grained multimodal pretraining framework that integrates protein sequences, local protein 3D structural regions, and functional descriptions to ground functional phrases in the regions that support them. Specifically, we first construct ProtGround-Align, a large-scale dataset of fine-grained local region–function pairs. Building on this dataset, we propose ProtGround-Hier, a pretraining method that localizes the local protein regions supporting a functional phrase through joint modeling of protein–region–phrase relationships. It uses multi-positive region–phrase alignment, residue–token optimal transport, and multi-level masked modeling. Extensive experiments show that ProtGround consistently outperforms strong baselines on fine-grained grounding and protein understanding tasks. These results show that grounding functional language in local regions enables function-aware, spatially localized protein understanding and improves transfer across tasks. The data and code will be made publicly available.

1 INTRODUCTION

Protein functions often depend on local 3D environments formed by residues that may be distant in sequence but become spatially proximal upon folding (Laskowski et al., 2005). We refer to these functionally relevant residue sets as *local protein 3D structural regions*. A protein can contain multiple such regions supporting different functions (Wu et al., 2010). Identifying the local regions is critical for revealing molecular mechanisms, interpreting disease-associated mutations, discovering druggable sites, and guiding protein engineering. Therefore, understanding protein function requires determining not only what a protein does, but also where each function is realized in its 3D structure.

Protein modeling has advanced substantially in recent years. Protein language models pretrained on large-scale protein data have achieved strong performance in protein understanding, structural modeling, and design (Lin et al., 2023; Chen et al., 2025; Hayes et al., 2025). Building on this progress, multimodal protein models further integrate natural-language and structure to complement protein sequences with explicit functional semantics and spatial information, improving performance on function prediction and cross-modal retrieval (Su et al., 2025; Hsieh et al., 2025). Despite these advances, most existing methods still operate at whole-protein level with global textual descriptions.

Such whole-protein modeling cannot localize specific functional descriptions to the local regions that support them, especially when a protein contains multiple regions with distinct biochemical roles. Figure 1(a) shows that whole-protein–text modeling captures MurA's overall enzymatic function but does not reveal which local regions support catalysis, PEP binding, and UDP-GlcNAc binding. Figure 1(b) shows how fine-grained grounding identifies which local region supports each functional phrase, while Figure 1(c) highlights why contiguous sequence windows are insufficient for local regions. However, learning these local region–function relationships poses three challenges. First, large-scale local structural region–functional phrase pairs are scarce. Second, these relationships are hierarchical and many-to-many, as proteins contain multiple regions, regions can support multiple phrases, and similar phrases can recur across regions and proteins. Third, grounding must model relationships between functions and structural evidence at both the region and residue levels.

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.

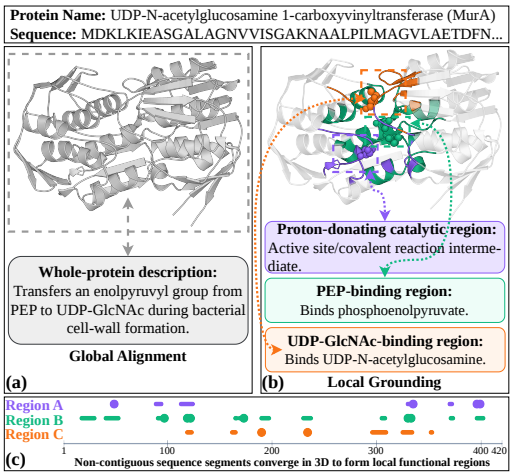

Figure 1: From whole-protein modeling to fine-grained functional grounding. **(a)** Whole-protein–text modeling captures global protein function. **(b)** ProtGround grounds functional phrases in local regions. **(c)** Grounded regions contain non-contiguous sequence residues that converge in 3D.

To address the data challenge, we construct ProtGround-Align, a large-scale dataset of fine-grained local region–function pairs, through two complementary routes. In the direct route, we derive feature-specific functional phrases and their residue positions from curated Swiss-Prot residue-level feature annotations, map these positions to protein structures from AlphaFold DB or RCSB PDB, and construct the corresponding local 3D structural regions. Since each phrase and region originate from the same feature annotation, they directly form high-confidence direct pairs. In the semantic route, we broaden functional coverage by decomposing protein-level descriptions into fine-grained phrases with a multi-stage LLM harness, combining them with GO and other curated annotations, and matching the resulting candidate phrases to local regions using SapBERT with similarity and ligand-consistency filtering. An independent LLM validator then retains only reliable semantic pairs. Overall, ProtGround-Align contains 2,939,441 local region–function pairs and captures hierarchical and many-to-many relationships among proteins, local regions, and functional phrases.

To this end, we propose ProtGround-Hier, a pretraining method that localizes the protein regions supporting a functional phrase by modeling hierarchical many-to-many relationships. At the region level, multi-positive local region–phrase alignment uses contrastive learning to align each local region with all associated functional phrases. At the residue level, residue–token optimal transport models soft correspondences between regional residues and phrase tokens. Multi-level masked modeling further strengthens cross-modal alignment while preserving masked protein and text modeling. Extensive experiments on fine-grained protein grounding and protein understanding tasks show that ProtGround consistently outperforms strong baselines and captures fine-grained cross-modal correspondences and localized functional knowledge for spatially grounded protein understanding.

Our contributions are threefold. First, we construct ProtGround-Align, a large-scale dataset of fine-grained functional phrase–local 3D region pairs for protein grounding. Second, we propose ProtGround-Hier, a pretraining method for fine-grained grounding that models hierarchical and many-to-many relationships. Third, extensive experiments demonstrate ProtGround's effectiveness.

2 ProtGround-Align: Dataset Construction

Existing protein–text datasets mainly pair whole proteins with global textual descriptions. As summarized in Table 1, they cannot localize specific functional phrases to the local regions that support them. To address these limitations, we construct ProtGround-Align through two routes, as shown in Figure 2. The direct route derives high-confidence region–function pairs from residue-level annotations, while the semantic route broadens functional coverage by constructing fine-grained candidate phrases from protein-level descriptions and curated annotations, then matching them to local regions with semantic and consistency filtering followed by independent validation to obtain semantic pairs.

Table 1: Comparison of multimodal protein–text datasets.

Dataset	Granularity	Modality	Scale (M)
ProtDescribe (Xu et al., 2023)	Global	Sequence + Text	0.553
SwissProtCLAP (Liu et al., 2025)	Global	Sequence + Text	0.441
Prot2Text-Data (Abdine et al., 2024)	Global	Sequence + 3D Structure + Text	0.257
InterPT (Zhuo et al., 2024)	Global	Sequence + Interleaved Text	0.429
ProTrek (Su et al., 2025)	Global	Sequence + 3D Structure + Text	40.0
ProteinAligner (Zhang et al., 2026)	Global	Sequence + 3D Structure + Text	0.134
OPUS-InstructionCorpus (Lv et al., 2026)	Global	Sequence + Instruction Text	2.110
ProtGround-Align	**Local**	**Sequence + Local 3D Structure + Text**	**2.939**

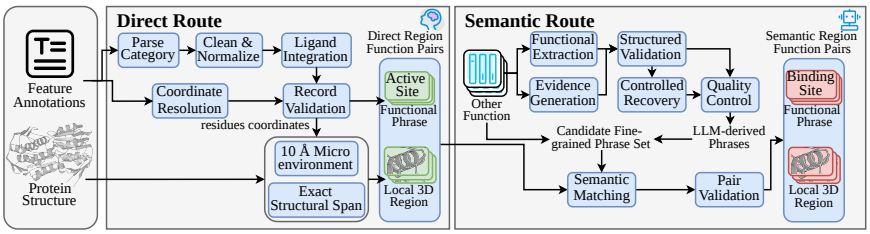

Figure 2: The ProtGround-Align construction pipeline, including the direct route and semantic route.

2.1 Direct Region–Function Pair Construction

We construct direct region–function pairs from curated residue-level feature annotations in Swiss-Prot (Bairoch & Apweiler, 2000). For each annotation, we parse the feature type to identify the functional category, clean and normalize the curated description to remove irrelevant content and preserve essential functional information, incorporate structured ligand information for molecular specificity, and resolve the annotated residue coordinates to construct a feature-specific functional phrase with an explicit residue or residue span. We then validate the functional phrase–residue coordinates record and retain it only when the functional content is eligible and the residue coordinates are exact, valid, and localizable. We then obtain the corresponding protein structure from AlphaFold DB (Varadi et al., 2022) or RCSB PDB (Berman et al., 2000) and map the annotated residue or residue span onto the structure. For localized residue positions, we define the local 3D structural region as the 10 Å microenvironment containing residues within a minimum interatomic distance of 10 Å from any annotated residue; for residue spans, we use the exact mapped structural span. Since each feature-specific functional phrase and local region are derived from the same residue-level annotation, they are directly paired. Finally, we retain only pairs with consistent sequence–region mapping and valid structural geometry, yielding high-confidence direct region–function pairs.

2.2 Semantic Region–Function Pair Construction

Although residue-level feature annotations yield high-confidence direct pairs, they capture only part of each protein's functional information. We therefore collect other diverse functional information from Swiss-Prot, including protein-level functional descriptions, GO annotations, keywords, similarity and family annotations, and curated information on cofactors, catalytic activities, activity regulation, subunit organization, and post-translational modifications, etc. However, protein-level descriptions are not fine-grained, and these sources do not directly specify the corresponding local 3D regions. To broaden fine-grained functional coverage, we construct semantic pairs in three stages. Fine-Grained Functional Phrase Generation uses a controlled multi-stage LLM harness to decompose protein-level descriptions into validated fine-grained functional phrases and combines them with GO and other curated annotations into a candidate phrase set. Semantic Matching to Local 3D Structural Regions uses SapBERT with similarity and ligand-consistency filtering to match the candidate phrases to the constructed local regions. Independent Pair Validation then uses a separate LLM validator to verify the resulting semantic matches and retain only reliable semantic pairs.

Fine-Grained Functional Phrase Generation. Protein-level descriptions often combine multiple distinct functional facts, such as catalytic activity, ligand binding, and regulatory roles, and therefore require decomposition into fine-grained phrases. We process each description using a Qwen3.5-based (Team, 2026) multi-stage LLM harness with four controlled stages.

1. Target-Aware Functional Extraction. The harness first retrieves protein identity metadata for each functional description and retains only functions attributable to the target protein, excluding content about other proteins, molecules, or pathways. The retained content is then decomposed into LLM-derived fine-grained phrases under a one-function-per-phrase rule while preserving essential context, including substrates, products, targets, and uncertainty qualifiers. *2. Evidence-Constrained Structured Generation.* For each extracted phrase, the harness simultaneously generates supporting evidence from the source information under a predefined structured schema, ensuring that each LLM-derived phrase is supported by source evidence. *3. Structured Output Validation and Controlled Recovery.* The generated phrase–evidence pairs are validated for consistency with the structured schema and source information. If primary extraction yields no valid phrase, conservative recovery stages re-examine the description to recover only explicit target-specific functions. *4. Deduplication and Multi-Stage Quality Control.* The LLM-derived fine-grained phrases from the primary and recovery stages are merged, deduplicated, and filtered to remove unsupported, negative-only, non-functional, or incorrectly attributed outputs while preserving essential functional context. The resulting phrases are concise, self-contained and fine-grained, each expressing a single function, such as catalytic activity, cofactor usage, or binding, etc, while preserving substrates, products, targets, uncertainty qualifiers, and other context required to maintain its functional meaning. Details of harness are provided in Appendix A.1. The validated LLM-derived phrases are then combined with GO and other curated annotations to form candidate fine-grained phrase set for semantic matching.

Semantic Matching to Local 3D Structural Regions. For each candidate fine-grained phrase, we restrict semantic matching to local regions constructed from residue-level annotations of the same protein. We encode each candidate phrase and the feature-specific functional phrase derived from the residue-level annotation used to construct each local region with SapBERT (Liu et al., 2021), and compute their semantic similarity using cosine similarity to identify semantically consistent region–phrase matches. We retain only matches with similarity above 0.80 and a margin of at least 0.20 over the next-best region. For ligand-related fine-grained phrases, we further require the ligand specified in the candidate phrase to match that in the feature-specific functional phrase of the local region. The retained matches form semantic region–function pairs for independent pair validation.

Independent Pair Validation. We use GLM-5.3 (Zeng et al., 2026) as an independent LLM validator, separate from the phrase-generation harness, to provide a second quality-control after semantic matching. For each pair, the validator compares functional phrase with curated residue-level feature annotation of the matched local region to assess functional consistency. Ambiguous, unsupported, or functionally inconsistent pairs are rejected, retaining reliable semantic region–function pairs.

2.3 ProtGround-Align Statistics and Insights

ProtGround-Align contains 2,939,441 local region–function pairs spanning 376,886 proteins and 2,110,831 unique local 3D structural regions. The direct route contributes 2,224,693 pairs (75.68%), while semantic route contributes 714,748 (24.32%). Direct pairs are almost entirely derived from curated Swiss-Prot feature annotations, whereas semantic pairs incorporate diverse functional sources, including keywords (7.72%), LLM-derived phrases (7.21%), GO annotations (6.29%), cofactor-related annotations (2.94%), and post-translational modification annotations (0.15%). LLM-derived phrases contribute 45,663 unique phrases, further enriching functional diversity. The semantic route adds new functional phrases for 237,866 proteins (63.11%), doubling the median phrase count from 3 to 6 and increasing unique phrase–protein associations by 69.06%. Together, the two routes provide reliable local region–function pairs with broader functional coverage.

ProtGround-Align exhibits hierarchical and many-to-many relationships among proteins, local 3D structural regions, and functional phrases. On average, each protein contains 5.60 local 3D structural regions and is associated with 4.64 unique functional phrases; 81.72% and 80.06% of proteins have multiple regions and phrases, respectively. At the local-region level, 401,641 regions (19.03%) are associated with multiple unique functional phrases. Conversely, 45.52% of unique functional phrases occur across multiple local 3D structural regions, and 44.41% across multiple proteins.

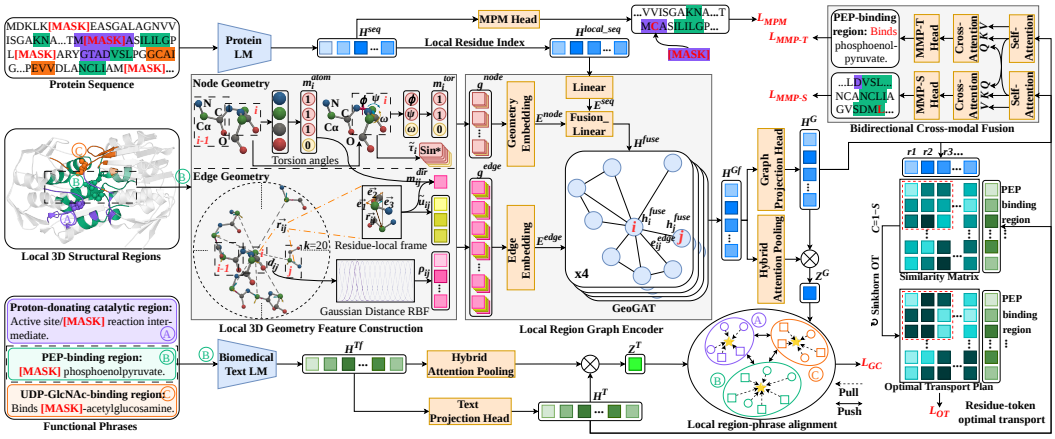

Figure 3: Overview of ProtGround-Hier. It learns fine-grained protein–function grounding.

3 ProtGround-Hier

3.1 Problem Formulation

We study fine-grained grounding of functional phrases in local protein 3D structural regions. Given ProtGround-Align dataset $\mathcal{D}$, each protein P comprises an amino acid sequence $S = [s_1, s_2, \ldots, s_n]$ of length n and M local regions $\mathcal{R} = \{R_1, R_2, \ldots, R_M\}$. Each region R_r contains Q_r residues and supports K phrases $\mathcal{T}_r = \{T_{r1}, T_{r2}, \ldots, T_{rK}\}$, where each phrase $T_{rk} = [t_1, t_2, \ldots, t_L]$ contains L text tokens. All phrases in $\mathcal{T}_r$ are treated as positives for R_r. Our goal is to localize the local protein regions that support a functional phrase by modeling hierarchical many-to-many relationships.

3.2 Model Architecture

Protein LM and Biomedical Text LM. As shown in Figure 3, we use a pre-trained ESM-2 as the Protein LM. Given a protein sequence S, ESM-2 produces residue representations $H^{seq} = [h_1^{seq}, \ldots, h_n^{seq}]$. For each local 3D region R_r containing Q_r residues, we gather their corresponding representations from H^{seq} to obtain H^{local_seq}. For each region–phrase pair (R_r, T_{rk}), we use a pre-trained SapBERT as the Biomedical Text LM to encode T_{rk} into token representations H^{Tf}.

Local 3D Geometry Feature Construction. For structure, we construct node and edge geometry features for local region R_r from backbone coordinates $\{x_i^N, x_i^{C_\alpha}, x_i^C, x_i^O\}_{i=1}^{Q_r}$ of the residues. Node geometry captures local backbone conformation of individual residues, while edge geometry captures pairwise spatial relations between neighboring residues. *1. Node Geometry.* For each residue i, we characterize its local backbone conformation using torsion angles ϕ_i, ψ_i, and ω_i, encoded as $\tau_i = [\sin\phi_i, \cos\phi_i, \sin\psi_i, \cos\psi_i, \sin\omega_i, \cos\omega_i]$. Invalid torsions are handled by a validity mask $m_i^{tor} = [m_i^\phi, m_i^\psi, m_i^\omega]$, where each entry indicates whether the required backbone atoms are present and geometrically valid. Duplicating each entry to match τ_i yields $\bar{m}_i^{tor}$, which is applied element-wise as $\widetilde{\tau}_i = \bar{m}_i^{tor} \odot \tau_i$. We also define an atom-availability mask $m_i^{atom} = [m_i^N, m_i^{C_\alpha}, m_i^C, m_i^O]$, indicating valid backbone coordinates. The final node geometry is $g_i^{node} = [\widetilde{\tau}_i; m_i^{tor}; m_i^{atom}]$. *2. Edge Geometry.* We characterize pairwise spatial relations between neighboring residues using distance and relative direction. For each residue i, we define $\mathcal{N}_i$ as its 20 nearest residues by C_α Euclidean distance $d_{ij} = \|x_j^{C_\alpha} - x_i^{C_\alpha}\|_2$, where $j \in \mathcal{N}_i$ denotes a neighboring residue, and construct a residue-local frame $\mathbf{F}_i$ from its N, C_α, and C atoms (Appendix A.2). We then express the relative C_α position from i to j in the local frame of i as $r_{ij} = \mathbf{F}_i^\top (x_j^{C_\alpha} - x_i^{C_\alpha})$, yielding the relative direction $u_{ij} = r_{ij}/\|r_{ij}\|_2$. Invalid directions are handled by a direction-validity mask m_{ij}^{dir}, requiring valid C_α coordinates for both residues and a nondegenerate local frame at residue i formed by its N, C_α, and C atoms with nonzero, non-collinear backbone directions; the masked direction is $\widetilde{u}_{ij} = m_{ij}^{dir} u_{ij}$. For distance, to obtain a smooth multi-scale representation that captures different distance ranges, we use $B = 16$ evenly spaced radial centers $\{\mu_b\}_{b=1}^B$ and expand d_{ij} with

Gaussian radial basis functions (RBFs) as $\rho_{ij,b} = \exp[-((d_{ij} - \mu_b)/\sigma)^2]$, $b = 1, \ldots, B$, yielding $\rho_{ij} = [\rho_{ij,1}, \ldots, \rho_{ij,B}]$. The final edge geometry is $g_{ij}^{edge} = [\rho_{ij}; \widetilde{u}_{ij}; m_{ij}^{dir}]$. Aggregating over all residues and edges yields $g^{node} = \{g_i^{node}\}_{i=1}^{Q_r}$ and $g^{edge} = \{g_{ij}^{edge}\}_{i=1,\ldots,Q_r,\ j\in\mathcal{N}_i}$.

Local Region Graph Encoder. The encoder takes H^{local_seq}, g^{node}, and g^{edge} as inputs. Geometry Embedding maps node geometry g^{node} to $E^{node} = [e_1^{node}, \ldots, e_{Q_r}^{node}]$, while Edge Embedding maps g^{edge} to E^{edge}. We linearly project local sequence features H^{local_seq} to the node geometry dimension as $E^{seq} = [e_1^{seq}, \ldots, e_{Q_r}^{seq}]$. For each residue i, a feature-wise gate $\gamma_i = \sigma(W_g[e_i^{seq}; e_i^{node}])$ fuses the sequence and node geometry features, yielding $H^{fuse} = [h_1^{fuse}, \ldots, h_{Q_r}^{fuse}]$, with

$$h_i^{fuse} = \mathrm{LN}\big(f_{\mathrm{fuse}}([e_i^{seq}; e_i^{node}]) + \gamma_i \odot e_i^{seq} + (1-\gamma_i) \odot e_i^{node}\big). \tag{1}$$

We encode local region with a four-layer GeoGAT (Veličković et al., 2018), initializing node states as $h_i^{(0)} = h_i^{fuse}$. At layer ℓ, the message and attention for neighbor $j \in \mathcal{N}_i$ are $m_{ji}^{(\ell)} = f_m^{(\ell)}([\mathrm{LN}(h_j^{(\ell)}); e_{ij}^{edge}])$ and $\alpha_{ji}^{(\ell)} = \underset{j\in\mathcal{N}_i}{\mathrm{softmax}} f_a^{(\ell)}([\mathrm{LN}(h_i^{(\ell)}); \mathrm{LN}(h_j^{(\ell)}); e_{ij}^{edge}])$. The messages are aggregated with attention and residual feed-forward updates, yielding $H^{Gf} = [h_1^{Gf}, \ldots, h_{Q_r}^{Gf}]$:

$$\hat{h}_i^{(\ell)} = h_i^{(\ell)} + W_o^{(\ell)} \sum_{j\in\mathcal{N}_i} \alpha_{ji}^{(\ell)} \odot m_{ji}^{(\ell)}, \qquad h_i^{(\ell+1)} = \hat{h}_i^{(\ell)} + \mathrm{FFN}^{(\ell)}(\mathrm{LN}(\hat{h}_i^{(\ell)})). \tag{2}$$

Subsequently, the Graph and Text Projection Heads P_G and P_T map the graph and text features into a shared protein–text space, yielding $H^G = P_G(H^{Gf})$ and $H^T = P_T(H^{Tf})$. Hybrid Attention Pooling then computes residue and token importance weights from H^{Gf} and H^{Tf}, respectively, and pools the projected features H^G and H^T into z^G and z^T. For the graph branch, the residue weights are $\alpha_i = \frac{\exp(f_G(h_i^{Gf}))}{\sum_{q=1}^{Q_r}\exp(f_G(h_q^{Gf}))}$, and the pooled representation is $z^G = \mathrm{Norm}\Big(\sum_{i=1}^{Q_r} \alpha_i h_i^G\Big)$.

3.3 Pre-training Objectives

Local Region–Phrase Alignment. At the region level, we use multi-positive contrastive learning to align R_r with all phrases in $\mathcal{T}_r$ as positives. The region–phrase similarity is computed by cosine similarity, $s_{rk} = \cos_\mathrm{sim}(z_r^G, z_{rk}^T)$. Given batch regions $\mathcal{B}_G$ and phrases $\mathcal{B}_T$, we compute region-to-phrase loss $\mathcal{L}_{G\to T}$ and symmetric phrase-to-region loss $\mathcal{L}_{T\to G}$, yielding final alignment loss:

$$\mathcal{L}_{G\to T} = -\frac{1}{|\mathcal{B}_G|}\sum_{r\in\mathcal{B}_G}\frac{1}{|\mathcal{T}_r|}\sum_{T_{rk}\in\mathcal{T}_r}\log\frac{e^{s_{rk}}}{\sum_{k'\in\mathcal{B}_T} e^{s_{rk'}}}, \qquad \mathcal{L}_{GC} = \frac{\mathcal{L}_{G\to T} + \mathcal{L}_{T\to G}}{2}. \tag{3}$$

Residue–Token Optimal Transport. At the residue level, we use optimal transport to model soft correspondences between regional residues and phrase tokens, assigning larger transport mass to stronger correspondences without explicit residue–token annotations. We construct a residue–token similarity matrix $S \in \mathbb{R}^{Q_r\times L}$ with $S_{ij} = \cos_\mathrm{sim}(h_i^G, h_j^T)$ and define the transport cost as $C_{ij} = 1 - S_{ij}$. Let a and b denote uniform marginals over regional residues and phrase tokens, and $\mathcal{U}(a,b)$ the corresponding transport polytope. The Sinkhorn transport plan (Cuturi, 2013) and OT loss are

$$\Pi^* = \arg\min_{\Pi\in\mathcal{U}(a,b)} \langle \Pi, C\rangle + \varepsilon\sum_{i,j}\Pi_{ij}(\log\Pi_{ij} - 1), \qquad \mathcal{L}_{OT} = \langle \Pi^*, C\rangle = \sum_{i,j}\Pi_{ij}^* C_{ij}. \tag{4}$$

Multi-level Masked Modeling. Finally, multi-level masked modeling combines masked protein modeling with bidirectional cross-modal masked modeling. The superscript m denotes representations obtained from masked inputs through the same encoders and projection heads. For masked protein modeling, the MPM Head predicts masked amino acids from $H^{seq,m}$ at positions $\mathcal{M}_P$:

$$\mathcal{L}_{MPM} = -\frac{1}{|\mathcal{M}_P|}\sum_{i\in\mathcal{M}_P}\log p(s_i \mid h_i^{seq,m}). \tag{5}$$

For cross-modal masked modeling, $H^{G,m}$ and $H^{T,m}$ undergo self-attention followed by bidirectional cross-attention, yielding $\widetilde{H}^{G,m}$ and $\widetilde{H}^{T,m}$. The MMP-S and MMP-T Heads predict masked residues and phrase tokens at positions $\mathcal{M}_G$ and $\mathcal{M}_T$, respectively, with

$$\mathcal{L}_{MMP} = -\frac{1}{2}\Big[\frac{1}{|\mathcal{M}_G|}\sum_{i\in\mathcal{M}_G}\log p(s_i \mid \widetilde{h}_i^{G,m}) + \frac{1}{|\mathcal{M}_T|}\sum_{j\in\mathcal{M}_T}\log p(t_j \mid \widetilde{h}_j^{T,m})\Big]. \tag{6}$$

Here, the residue- and phrase-prediction terms correspond to $\mathcal{L}_{MMP-S}$ and $\mathcal{L}_{MMP-T}$, respectively.
Overall Pre-training Objective. During pre-training, we minimize the losses of all objectives:

$$\mathcal{L} = \mathcal{L}_{GC} + \lambda_{OT}\mathcal{L}_{OT} + \alpha_{MPM}\mathcal{L}_{MPM} + \beta_S\mathcal{L}_{MMP-S} + \beta_T\mathcal{L}_{MMP-T}. \tag{7}$$

where λ_{OT}, α_{MPM}, β_S, and β_T control the contributions of the corresponding objectives.

4 Experiments and Analysis

4.1 Pre-training Settings

ProtGround-Hier is pre-trained on ProtGround-Align for 30 epochs with AdamW (Loshchilov & Hutter, 2017) (weight decay 0.01; learning rates 10^{-5}, 2×10^{-5}, and 10^{-4} for ESM-2, SapBERT, and newly initialized modules). We set $\lambda_{OT} = 0.1$, $\alpha_{MPM} = 1.0$, $\beta_S = 1.0$, and $\beta_T = 1.0$.

4.2 Main Results

4.2.1 Fine-Grained Protein Grounding

Evaluation Benchmark. To evaluate fine-grained protein-function grounding on unseen proteins, we first collect proteins that are absent from ProtGround-Align. These proteins are further filtered against those in ProtGround-Align by sequence similarity, measured with MMseqs2 (Steinegger & Söding, 2017) as the fraction of identical residues over the alignment length, to form two subsets with increasing difficulty. *U50* contains proteins with at most 50% sequence identity, providing a broad test on sequence-divergent proteins. *U30-hard* further lowers this cutoff to 30%, excluding close homologs and retaining more distantly related proteins, thereby making fine-grained grounding substantially more challenging. Based on these two subsets, we construct *ProtGround-Unseen* using the ProtGround-Align construction pipeline, where only the direct route is used to ensure high annotation reliability, yielding high-confidence region–function pairs for evaluating grounding.

Baselines. We compare ProtGround-Hier with multimodal baselines, including ProteinCLAP (Liu et al., 2025) and ProtCLIP (Zhou et al., 2025), which ground functions from sequence and text only, and ProTrek (Su et al., 2025), which jointly models sequence, structure, and text.

Tasks and Metrics. We evaluate grounding at three levels of increasing localization difficulty. *Level 1: Protein Retrieval* identifies the proteins associated with a functional phrase from a candidate protein set. *Level 2: Candidate Grounding* further requires localizing the regions supporting a phrase among candidate regions within a protein. *Level 3: Blind Grounding* does not rely on candidate regions and directly localizes the supporting regions across the whole protein structure. We evaluate Levels 1 and 2 using mean average precision (mAP), which measures the extent to which ground-truth target are ranked near the top, and Hit@1, which measures whether the ground-truth target is ranked first. Level 3 uses Core mAP to measure how highly true core residues are ranked, and SiteRecall@5 to measure whether the top-5 predicted regions recover the true functional sites.

Experimental Results. Table 2 shows that ProtGround-Hier consistently outperforms all baselines across the three levels on both *U50* and *U30-hard*. The advantage remains substantial on the more challenging *U30-hard* and is particularly pronounced for blind grounding. These results demonstrate that ProtGround-Hier can identify proteins associated with a functional phrase and accurately localize the regions that support each phrase, even in previously unseen proteins.

Case Studies. Figure 4 presents four representative cases on *ProtGround-Unseen*. ProtGround-Hier accurately localizes the 3D regions supporting each functional phrase and simultaneously pinpoints the corresponding core-site residues, achieving a SiteRecall@5 of 1.00. In Fig. 4(a), the localized Mg^{2+}-, proton-acceptor-, and diphosphate-related regions together resolve the catalytic pocket, helping reveal the underlying molecular mechanism. In Fig. 4(b), the localized chloride-, FAD-, and catalytic-lysine regions provide precise functional sites that can guide protein engineering. In Fig. 4(c), the localized DNA–tyrosine intermediate and catalytic arginine regions provide residue-level structural context for interpreting functionally important mutations. In Fig. 4(d), the localized substrate-binding and catalytic regions illustrate the potential for discovering functionally relevant druggable sites. These cases demonstrate the practical value of fine-grained grounding for mechanistic understanding, mutation interpretation, druggable-site discovery, and protein engineering.

Table 2: Performance comparison on protein retrieval and grounding in *ProtGround-Unseen*.

	Level1: Protein Retrieval		Level2: Candidate Grounding		Level3: Blind Grounding	
Model	**mAP ↑**	**Hit@1 ↑**	**mAP ↑**	**Hit@1 ↑**	**Core mAP ↑**	**SiteRecall@5 ↑**
U50						
ProteinCLAP Liu et al. (2025)	0.4133	0.3957	0.6161	0.6045	0.2183	0.3664
ProtCLIP Zhou et al. (2025)	0.4272	0.4151	0.6439	0.6246	0.2252	0.3826
ProTrek Su et al. (2025)	0.4985	0.4793	0.6988	0.6751	0.3084	0.5173
ProtGround-Hier	**0.7463**	**0.7176**	**0.9334**	**0.9112**	**0.5618**	**0.8692**
U30-hard						
ProteinCLAP Liu et al. (2025)	0.3721	0.3437	0.5892	0.5518	0.1868	0.3606
ProtCLIP Zhou et al. (2025)	0.3874	0.3546	0.5974	0.5796	0.2061	0.3783
ProTrek Su et al. (2025)	0.4352	0.4059	0.6615	0.6375	0.2670	0.4916
ProtGround-Hier	**0.6985**	**0.6386**	**0.8958**	**0.8644**	**0.5129**	**0.8622**

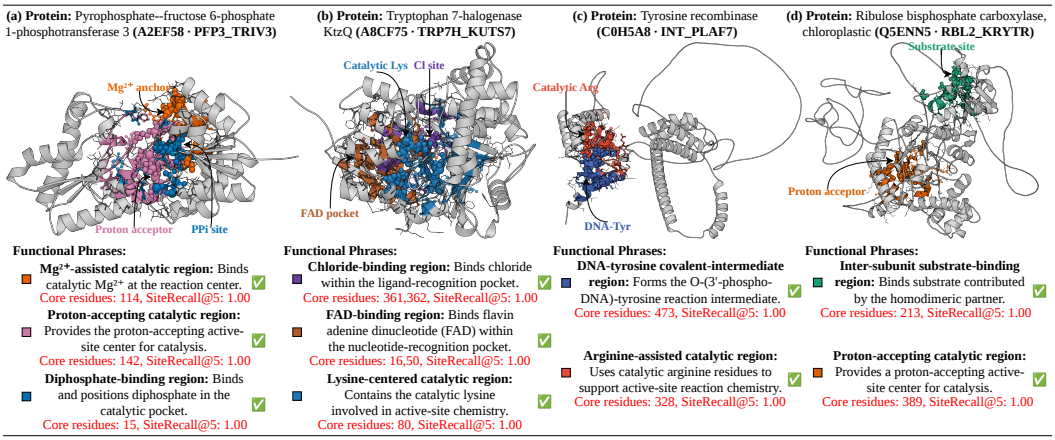

Figure 4: Case studies of fine-grained protein–function grounding on *ProtGround-Unseen*.

4.2.2 Protein Understanding Tasks

Downstream Benchmark Tasks. We evaluate ProtGround-Hier on 11 downstream tasks covering protein localization, fitness prediction, and function annotation. For localization, we use binary and subcellular localization tasks from DeepLoc (Almagro Armenteros et al., 2017). For fitness prediction, we evaluate on β-lactamase, AAV, thermostability, fluorescence, and stability benchmarks from PEER, FLIP, and TAPE (Xu et al., 2022; Dallago et al., 2021; Rao et al., 2019). For function annotation, we consider Enzyme Commission (EC) prediction and the three Gene Ontology (GO) prediction branches, including biological process (GO-BP), molecular function (GO-MF), and cellular component (GO-CC), following the DeepFRI benchmarks (Gligorijević et al., 2021).

Baselines. We compare ProtGround-Hier with sequence encoders trained from scratch, including CNN (Shanehsazzadeh et al., 2020), ResNet, LSTM, and Transformer (Rao et al., 2019), as well as PLMs baselines, including OntoProtein (Zhang et al., 2022), ESM-1b (Rives et al., 2021), ESM-2 (Lin et al., 2023), ProtST (Xu et al., 2023) and ProtCLIP (Zhou et al., 2025).

Training and Evaluation. We evaluate all PLMs under both fixed-encoder learning and full-model tuning. We train localization and fitness models for 100 epochs and function annotation models for 50 epochs. We report accuracy for localization, Spearman's ρ for fitness prediction, and AUPR and F_{max} for function annotation. Training details and metric definitions are provided in Appendix A.3.

Experimental Results. Table 3 shows that ProtGround-Hier improves localization, fitness prediction, and function annotation. Under fixed-encoder, it performs best on both localization and raises the mean Spearman's ρ for fitness prediction from 0.615 to 0.674 over ProtCLIP. Under full-model, it achieves the best binary localization accuracy and improves the mean fitness correlation from 0.793 to 0.813. For function annotation, the mean F_{max} increases from 0.609 to 0.633 with fixed encoders and from 0.679 to 0.696 with full-model. These results demonstrate the effectiveness.

Table 3: Performance comparison on localization, fitness prediction, and function annotation.

Models	**Loc class** (Acc.%)		**Fitness pred** (Spearman's ρ)					**Function anno** (AUPR & F_{max})							
	Bin	**Sub**	**β-lac**	**AAV**	**Thermo**	**Flu**	**Sta**	**EC**		**GO-BP**		**GO-MF**		**GO-CC**	
Protein sequence encoders trained from scratch															
CNN	82.67	58.73	0.781	0.746	0.494	0.682	0.637	0.540	0.545	0.165	0.244	0.380	0.354	0.261	0.387
ResNet	78.99	52.30	0.152	0.739	0.528	0.636	0.126	0.137	0.187	0.166	0.280	0.281	0.267	0.266	0.403
LSTM	88.11	62.98	0.139	0.125	0.564	0.494	0.533	0.032	0.082	0.130	0.248	0.100	0.166	0.150	0.320
Transformer	75.74	56.02	0.261	0.681	0.545	0.643	0.649	0.187	0.219	0.135	0.257	0.172	0.240	0.170	0.380
PLMs w/ fix-encoder learning															
OntoProtein	84.87	68.34	0.471	0.217	0.605	0.432	0.688	0.411	0.417	0.243	0.345	0.418	0.383	0.346	0.465
ESM-1b	91.61	79.82	0.528	0.454	0.674	0.430	0.750	0.649	0.642	0.309	0.403	0.557	0.528	0.404	0.504
ESM-2	91.32	80.84	0.559	0.374	0.677	0.456	0.746	0.711	0.694	0.311	0.412	0.577	0.547	0.404	0.519
ProtST	92.52	83.39	0.565	0.398	0.681	0.499	0.776	0.810	0.784	0.358	0.458	0.643	0.601	0.451	0.546
ProtCLIP	94.39	83.65	0.565	0.532	0.682	0.503	0.795	0.818	0.786	0.427	0.483	0.650	0.617	0.493	0.551
Ours	**94.62**	**85.17**	**0.683**	**0.533**	**0.726**	**0.631**	**0.795**	**0.838**	**0.809**	**0.493**	**0.526**	**0.661**	**0.637**	**0.525**	**0.558**
PLMs w/ full-model tuning															
OntoProtein	92.47	77.59	0.757	0.791	0.662	0.630	0.731	0.854	0.841	0.284	0.436	0.603	0.631	0.300	0.441
ESM-1b	92.40	78.13	0.839	0.821	0.669	0.679	0.694	0.884	0.869	0.332	0.452	0.630	0.659	0.324	0.477
ESM-2	91.72	78.67	0.867	0.817	0.672	0.677	0.718	0.888	0.874	0.340	0.472	0.643	0.662	0.350	0.472
ProtST	92.52	80.22	0.879	0.825	0.682	0.682	0.738	0.898	0.878	0.342	0.482	0.647	0.668	0.364	0.487
ProtCLIP	95.08	**85.34**	0.884	**0.892**	0.686	0.685	0.819	0.906	**0.908**	0.567	0.574	**0.696**	0.691	0.582	0.541
Ours	**95.22**	84.24	**0.917**	0.881	**0.713**	**0.717**	**0.838**	**0.916**	0.895	**0.587**	**0.598**	0.689	**0.705**	**0.603**	**0.586**

Table 4: Ablation study of pre-training objectives in ProtGround-Hier.

	Grounding				**Loc.** (Acc.%)		**Fit.** (ρ)		**Func.** (F_{max})
Config.	Level1 mAP	Level2 mAP	Level3 Core mAP	SiteRecall@5	Fixed	Full	Fixed	Full	
Full objective	**0.722**	**0.914**	**0.537**	**0.865**	**89.89**	**89.73**	**0.674**	**0.813**	**0.696**
w/o $\mathcal{L}_{\mathrm{GC}}$	0.694	0.868	0.487	0.819	87.44	88.01	0.647	0.790	0.658
w/o $\mathcal{L}_{\mathrm{OT}}$	0.682	0.841	0.462	0.797	87.86	88.36	0.646	0.787	0.654
w/o $\mathcal{L}_{\mathrm{MMP}}$	0.710	0.894	0.516	0.849	88.87	88.92	0.659	0.791	0.666
w/o $\mathcal{L}_{\mathrm{MPM}}$	0.703	0.885	0.508	0.840	88.72	88.74	0.662	0.793	0.664

4.3 Ablation Study

Table 4 shows that removing any pre-training objective degrades both fine-grained grounding and downstream protein understanding. For grounding, removing $\mathcal{L}_{\mathrm{OT}}$ causes the largest drop across all metrics, especially at Levels 2 and 3, highlighting the importance of residue–token correspondence for precise region localization. Removing $\mathcal{L}_{\mathrm{GC}}$ also substantially reduces performance, showing the role of region–phrase alignment, while removing $\mathcal{L}_{\mathrm{MMP}}$ or $\mathcal{L}_{\mathrm{MPM}}$ also causes performance drops. For downstream tasks, $\mathcal{L}_{\mathrm{GC}}$ most affects localization, while $\mathcal{L}_{\mathrm{OT}}$ most affects fitness prediction and function annotation. Overall, these results validate the effectiveness of our pre-training objectives.

5 Related Work

Protein language models perform strongly across protein understanding and design (Ismail et al., 2025). Building further, multimodal protein models further integrate text (Jararweh et al., 2025) and structure (Hsieh et al., 2025) to capture semantics and spatial information. However, these methods remain at whole-protein level and cannot localize functional descriptions to supporting local regions.

6 Conclusion and Future Work

We introduce ProtGround, a fine-grained multimodal framework for grounding functional phrases in local protein 3D structural regions. ProtGround-Align provides large-scale local region–function pairs, while ProtGround-Hier jointly models protein–region–phrase relationships to localize the protein regions supporting a functional phrase. Extensive experiments show that ProtGround improves fine-grained grounding and protein understanding tasks, highlighting its potential for functional-site discovery, drug design, mutation interpretation, and protein engineering. In the future, we will scale up dataset to improve generative capabilities and explore applications in protein design.

AI USE STATEMENT

Generative AI tools were used in this work for both research and writing assistance. As part of the ProtGround-Align construction pipeline, LLMs were used to generate fine-grained functional phrases from protein-level descriptions and to independently validate semantic region–function pairs. Generative AI tools were also used to assist with research methodology and experimental design, software development and debugging, literature search and summarization, result interpretation, and drafting and editing parts of the manuscript. We did not use generative AI to formulate or prove mathematical claims. All AI-assisted research outputs, code, analyses, and manuscript text were reviewed and verified by the authors, and all reported experimental results were obtained from the implemented methods and evaluated by the authors. We take full responsibility for the final content of this work.

REFERENCES

Hadi Abdine, Michail Chatzianastasis, Costas Bouyioukos, and Michalis Vazirgiannis. Prot2text: Multimodal protein's function generation with gnns and transformers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):10757–10765, 2024. doi: 10.1609/aaai.v38i10.28948.

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, 2017.

Amos Bairoch and Rolf Apweiler. The swiss-prot protein sequence database and its supplement trembl in 2000. *Nucleic acids research*, 28(1):45–48, 2000.

Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

Bo Chen, Xingyi Cheng, Pan Li, Yangli-ao Geng, Jing Gong, Shen Li, Zhilei Bei, Xu Tan, Boyan Wang, Xin Zeng, et al. xtrimopglm: unified 100-billion-parameter pretrained transformer for deciphering the language of proteins. *Nature Methods*, 22(5):1028–1039, 2025.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Christian Dallago, Jody Mou, Kadina E Johnston, Bruce J Wittmann, Nicholas Bhattacharya, Samuel Goldman, Ali Madani, and Kevin K Yang. Flip: Benchmark tasks in fitness landscape inference for proteins. *bioRxiv*, pp. 2021–11, 2021.

Vladimir Gligorijević, P Douglas Renfrew, Tomasz Kosciolek, Julia Koehler Leman, Daniel Berenberg, Tommi Vatanen, Chris Chandler, Bryn C Taylor, Ian M Fisk, Hera Vlamakis, et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):3168, 2021.

Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q Tran, Jonathan Deaton, Marius Wiggert, et al. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025.

Cheng-Yen Hsieh, Xinyou Wang, Daiheng Zhang, Dongyu Xue, Fei Ye, Shujian Huang, Zaixiang Zheng, and Quanquan Gu. Elucidating the design space of multimodal protein language models. *arXiv preprint arXiv:2504.11454*, 2025.

Aya Ismail, Tuomas Oikarinen, Amy Wang, Julius Adebayo, Samuel Stanton, Hector Corrada Bravo, Kyunghyun Cho, and Nathan Frey. Concept bottleneck language models for protein design. In *International Conference on Learning Representations*, volume 2025, pp. 56137–56169, 2025.

Ala Jararweh, Oladimeji Macaulay, David Arredondo, Yue Hu, Luis E Tafoya, Kushal Virupakshappa, and Avinash Sahu. Protein2Text: Resampling mechanism to translate protein sequences into human-interpretable text. In Weizhu Chen, Yi Yang, Mohammad Kachuee, and Xue-Yong Fu (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pp. 918–937, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-194-0. doi: 10.18653/v1/2025.naacl-industry.68. URL https://aclanthology.org/2025.naacl-industry.68/.

Roman A Laskowski, James D Watson, and Janet M Thornton. Protein function prediction using local 3d templates. *Journal of molecular biology*, 351(3):614–626, 2005.

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

Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. Self-alignment pretraining for biomedical entity representations. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 4228–4238, 2021.

Shengchao Liu, Yanjing Li, Zhuoxinran Li, Anthony Gitter, Yutao Zhu, Jiarui Lu, Zhao Xu, Weili Nie, Arvind Ramanathan, Chaowei Xiao, et al. A text-guided protein design framework. *Nature Machine Intelligence*, 7(4):580–591, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Ying Lv, Yifan Xu, Gang Xu, and Jianpeng Ma. Advancing generative large language models toward discriminative performance in protein function prediction. *Genome Biology*, 27(1):226, 2026.

Roshan Rao, Nicholas Bhattacharya, Neil Thomas, Yan Duan, Peter Chen, John Canny, Pieter Abbeel, and Yun Song. Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32, 2019.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the national academy of sciences*, 118(15):e2016239118, 2021.

Amir Shanehsazzadeh, David Belanger, and David Dohan. Is transfer learning necessary for protein landscape prediction? *arXiv preprint arXiv:2011.03443*, 2020.

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

Jin Su, Yan He, Shiyang You, Shiyu Jiang, Xibin Zhou, Xuting Zhang, Yuxuan Wang, Xining Su, Igor Tolstoy, Xing Chang, et al. A trimodal protein language model enables advanced protein searches. *Nature Biotechnology*, pp. 1–7, 2025.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, et al. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic acids research*, 50(D1):D439–D444, 2022.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Shirley Wu, Tianyun Liu, and Russ B Altman. Identification of recurring protein structure microenvironments and discovery of novel functional sites around cys residues. *BMC Structural Biology*, 10(1):4, 2010.

Minghao Xu, Zuobai Zhang, Jiarui Lu, Zhaocheng Zhu, Yangtian Zhang, Ma Chang, Runcheng Liu, and Jian Tang. Peer: a comprehensive and multi-task benchmark for protein sequence understanding. *Advances in Neural Information Processing Systems*, 35:35156–35173, 2022.

Minghao Xu, Xinyu Yuan, Santiago Miret, and Jian Tang. Protst: Multi-modality learning of protein sequences and biomedical texts. In *International conference on machine learning*, pp. 38749–38767. PMLR, 2023.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Li Zhang, Han Guo, Leah Schaffer, Young Su Ko, Digvijay Singh, Hamid Rahmani, Danielle Grotjahn, Elizabeth Villa, Michael Gilson, Wei Wang, et al. A tri-modal contrastive learning framework for protein representation learning. *Cell Reports Methods*, 6(5), 2026.

Ningyu Zhang, Zhen Bi, Xiaozhuan Liang, Siyuan Cheng, Haosen Hong, Shumin Deng, Jiazhang Lian, Qiang Zhang, and Huajun Chen. Ontoprotein: Protein pretraining with gene ontology embedding. *arXiv preprint arXiv:2201.11147*, 2022.

Hanjing Zhou, Mingze Yin, Wei Wu, Meng Li, Kai Fu, Jiacheng Chen, Jun Wu, and Zhen Wang. Protclip: Function-informed protein multi-modal learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(21):22937–22945, 2025. doi: 10.1609/aaai.v39i21.34456.

Le Zhuo, Zewen Chi, Minghao Xu, Heyan Huang, Jianan Zhao, Heqi Zheng, Conghui He, Xian-Ling Mao, and Wentao Zhang. ProtLLM: An interleaved protein-language LLM with protein-as-word pre-training. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8950–8963, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.484. URL https://aclanthology.org/2024.acl-long.484/.

A APPENDIX

A.1 LLM HARNESS FOR FINE-GRAINED FUNCTIONAL PHRASE GENERATION

We use a controlled multi-stage LLM harness to transform protein-level functional descriptions into fine-grained functional phrases. The harness is designed around a precision-first principle: functional information explicitly stated in the source annotation should be preserved, while unstated biological knowledge or inferred mechanisms should not be introduced. We use Qwen3.5-9B as the language model and constrain generation through structured prompts, target-aware attribution, JSON-Schema outputs, and automatic validation.

1. Target-Aware Functional Extraction. For each UniProtKB/Swiss-Prot FUNCTION description, the harness first retrieves the protein identity metadata associated with that description, including the entry name, protein name, primary accession, and molecule label when available. Based on this information, the harness retains only functional content explicitly attributable to the current target protein. Functions referring to other proteins, enzymes, receptors, molecules, pathways, or other biological actors mentioned in the description are excluded unless the source explicitly attributes those functions to the current target protein.

The retained target-specific functional content is then decomposed into fine-grained functional phrases following a one-function-per-phrase rule. Distinct functions are represented separately, while substrates, products, targets, reaction directions, uncertainty qualifiers, and other context required to preserve the meaning of each function remain attached to the corresponding phrase. This allows the harness to obtain fine-grained functional phrases without fragmenting information that is necessary to express a coherent function. The resulting phrases may describe molecular activities, catalytic functions, substrate or cofactor usage, binding functions, functional roles, biological-process contributions, or physiological effects, provided that the corresponding functional information is explicitly stated in the source description.

2. Evidence-Constrained Structured Generation. For each extracted fine-grained functional phrase, the harness simultaneously generates a supporting evidence span taken directly from the original functional description. More specifically, the evidence must be an exact contiguous substring of the source description that supports the corresponding phrase. The functional phrase and its evidence are produced under a predefined structured schema consisting of `phrase–evidence` pairs. This structured generation ensures that every LLM-derived phrase is explicitly based on the source description and retains a direct textual connection to the original annotation.

The prompt used for target-aware functional extraction and evidence-constrained structured generation is shown below. The corresponding protein identity metadata and UniProtKB/Swiss-Prot FUNCTION description are inserted into the designated placeholders at runtime.

Main prompt for fine-grained functional phrase extraction

```
You are an expert protein annotation curator.

CURRENT TARGET
entry_name: <ENTRY_NAME>
entry_short_name: <ENTRY_SHORT_NAME>
protein_name: <PROTEIN_NAME>
current_primary_accession: <PRIMARY_ACCESSION>
FUNCTION molecule label: <MOLECULE_LABEL>

TARGET OWNERSHIP CONTRACT
Apply the function-extraction rules below only to functions explicitly
     owned by or attributed to the CURRENT TARGET/current FUNCTION
    molecule or isoform. Other named proteins, genes, enzymes,
    receptors, substrates, pathways, cells, complexes, or actors may
    be context. Never transfer another actor's action to the current
    target. Use the target identity above only to resolve attribution;
     never use it to invent a function not stated by SOURCE.
```

```
Extract up to 32 distinct, concise, fine-grained protein function
    phrases that are explicitly supported by the provided protein
    function description.

The primary goal is to preserve as much explicitly stated functional
    information as possible without inventing information.

Each output phrase should express one coherent functional meaning that
     can stand on its own as a fine-grained protein function phrase.

A function phrase may describe a molecular activity, functional role,
    biological-process contribution, functional effect, physiological
    effect, functional classification, or another explicitly stated
    protein function.

Follow these rules strictly:
1. Use only functional information explicitly supported by the
    description. Never use external biological knowledge.

2. Never invent or infer an unstated mechanism, receptor, molecular
    partner, ligand, substrate, cofactor, pathway, binding event,
    catalytic event, activation event, inhibition event, interaction,
    structural site, or other biological fact.

3. Preserve all important explicitly stated information needed to
    express a function correctly.

4. Conciseness must never override information completeness. Do not
    shorten a phrase if doing so removes an explicitly stated
    substrate, product, target, reaction direction, functional effect,
     uncertainty qualifier, or other information required to preserve
    the complete functional meaning.

5. Prefer concise phrase-like wording, but make each phrase as long as
     necessary to express its function correctly and completely.

6. One phrase should represent one coherent functional meaning. Do not
     combine multiple independent functions merely because they occur
    in the same sentence.

7. Do not mechanically split every noun, argument, or qualifier into a
     separate phrase. Substrates, products, reaction directions,
    targets, and explicitly stated mechanistic qualifiers may remain
    together when they describe the same coherent function.

8. A modifier or clause that cannot stand alone as a protein function
    must not be emitted as a separate phrase. For example, a mechanism
     introduced by "via ..." should remain attached to the function it
     qualifies.

9. If the same functional activity is explicitly stated for multiple
    distinct named substrates, targets, ligands, products, or other
    independently groundable functional objects, those object-specific
     functions may be extracted as separate fine-grained phrases.

10. Different explicitly named substrates may therefore yield separate
     function phrases even when they share the same functional
    activity.

11. If the source provides both a broader class-level function and
    specific named members or examples, both may be retained when they
     provide useful distinct grounding labels.
```

```
12. Do not generate multiple paraphrases of the same functional fact.

13. Preserve uncertainty and epistemic qualifiers such as "may", "
    might", "probable", "putative", "likely", "possible", "potential",
     and "predicted". Never turn an uncertain function into a certain
    function.

14. Preserve functional context such as "in vitro" or "in macrophages"
     when removing it would change or overstate functional meaning.

15. The output represents positive or uncertainty-qualified functions
    that the protein is stated or suggested to possess, perform,
    contribute to, or participate in.

16. Do not output an explicitly negated function as a positive
    function phrase.

17. Every output phrase must itself contain functional meaning.

18. Never output a standalone phrase whose only meaning is location,
    occurrence, physical property, structural property, expression,
    taxonomy, or other non-functional metadata.

19. If a separable non-functional modifier can be removed without
    changing the functional meaning, remove it.

20. Functional classifications, biological roles, process
    contributions, physiological effects, and sensory effects are
    valid functions when explicitly stated. Do not convert them into a
     molecular mechanism unless that mechanism is explicitly provided.

21. Do not copy a long multi-clause source sentence when its
    independent functions can be represented more clearly as shorter
    function phrases.

22. Do not fill the phrase quota. Return only genuinely supported and
    useful function phrases.

23. When many distinct functions are present, prioritize distinct
    functional meanings and specific named functional objects over
    redundant paraphrases.

24. If no explicit functional information is present, return an empty
    list.

EVIDENCE CONTRACT
For every extracted function, return both: - phrase: the fine-grained
    functional phrase following the rules above; - evidence: an exact
    contiguous substring of SOURCE that explicitly supports that
    phrase.

Evidence is support and provenance. The phrase may be a faithful
    concise rendering, but it must not add information absent from its
     evidence or SOURCE.

Return only the JSON-Schema object requested by the caller. Do not
    output markdown, explanations, reasoning, comments, or text
    outside that object.

SOURCE
<SOURCE_FUNCTION_DESCRIPTION>
```

Structured output format

```
{
  "functions": [
    {
      "phrase": "<fine-grained functional phrase>",
      "evidence": "<exact contiguous supporting span from the original
     functional description>"
    }
  ]
}
```

3. Structured output validation and controlled recovery. The structured output from each generation pass is parsed automatically and checked against both the predefined schema and the original functional description. Each returned item must contain exactly one `phrase` and one `evidence` field, and every evidence span must occur verbatim in the source description. Malformed outputs, unexpected fields, missing phrases, or invalid evidence are not treated as valid extractions.

When the primary extraction yields no valid functional phrase, the harness activates a conservative recovery stage that re-evaluates the source under stricter rules and recovers only explicit positive functional information belonging to the current target protein. A second sparse recovery stage is used only for rare unresolved cases that still contain strong functional cues. Failures in request execution, structured parsing, or output validation are handled conservatively rather than being interpreted as valid empty annotations.

4. Deduplication and multi-stage quality control. Functional phrases obtained from the primary extraction and subsequent recovery stages are merged and exact duplicates are removed. The remaining phrases are then passed through successive functional-eligibility and quality-control stages. These stages remove unsupported, negative-only, non-functional, or incorrectly attributed phrases while preserving explicitly stated uncertainty and other qualifiers required to maintain the original functional meaning.

Phrases that remain unresolved after these controlled quality checks are quarantined rather than being treated as valid negative or empty examples. The validated LLM-derived phrases are finally combined with GO and other curated functional annotations before semantic matching to local 3D structural regions.

A.2 Residue-Local Frame Construction

For each center residue i, we construct a residue-local orthonormal frame from its backbone N, C_α, and C coordinates, with C_α as the frame origin. We first define the normalized N- and C-directed vectors as

$$e_{i1} = \frac{x_i^N - x_i^{C_\alpha}}{\|x_i^N - x_i^{C_\alpha}\|_2}, \qquad c_i = \frac{x_i^C - x_i^{C_\alpha}}{\|x_i^C - x_i^{C_\alpha}\|_2}. \tag{8}$$

The remaining orthogonal axes are constructed by

$$e_{i3} = \frac{e_{i1} \times c_i}{\|e_{i1} \times c_i\|_2}, \qquad e_{i2} = \frac{e_{i3} \times e_{i1}}{\|e_{i3} \times e_{i1}\|_2}, \tag{9}$$

yielding the residue-local frame

$$\mathbf{F}_i = [e_{i1}, e_{i2}, e_{i3}]. \tag{10}$$

For a neighboring residue j, its C_α displacement relative to the center residue i is expressed in the local frame of i as

$$r_{ij} = \mathbf{F}_i^\top \left(x_j^{C_\alpha} - x_i^{C_\alpha} \right), \qquad u_{ij} = \frac{r_{ij}}{\|r_{ij}\|_2}, \tag{11}$$

where u_{ij} represents the relative direction of residue j in the coordinate system centered at residue i.

A local frame is considered valid only when the N, C_α, and C coordinates are available and finite, the vectors used to construct the frame have nonzero norm, and the N- and C-directed vectors are non-collinear. These conditions, together with the validity of the C_α pair (i, j), determine the direction-validity indicator m_{ij}^{dir} used in the main text.

Table 5: Downstream training configurations. "Fixed" and "Full" denote fixed-encoder learning and full-model tuning, respectively. Fitness and function annotation use task-specific schedules; the epoch numbers denote the maximum training epochs.

Task	Setting	Optimizer	Learning Rate	Global Batch	Epochs
Localization	Fixed	Adam	$5e-5$	128	100
Localization	Full	AdamW	$5e-4$ / $1e-5$	16	100
β-lac	Fixed	AdamW	$2e-4$	64	100
AAV	Fixed	AdamW	$2e-4$	256	100
Thermostability	Fixed	AdamW	$2e-4$	64	100
Fluorescence	Fixed	AdamW	$2e-4$	64	100
Stability	Fixed	AdamW	$2e-4$	128	100
β-lac	Full	AdamW	$1e-5$	64	100
AAV	Full	AdamW	$1e-5$	128	100
Thermostability	Full	AdamW	$1e-5$	8	100
Fluorescence	Full	AdamW	$1e-5$	64	100
Stability	Full	AdamW	$1e-5$	64	100
EC	Full	AdamW	$1e-5$	8	50
GO-MF	Full	AdamW	$1e-5$	8	50
GO-BP	Full	AdamW	$1e-5$	8	50
GO-CC	Full	AdamW	$1e-5$	8	50

A.3 Downstream Training and Evaluation Details

Training Configurations. For downstream evaluation, we transfer the ProtGround-adapted ESM-2 encoder from the selected pre-training checkpoint. Localization and fitness prediction are evaluated under both fixed-encoder learning and full-model tuning, while function annotation uses full-model tuning. Since the benchmarks differ substantially in task structure and sequence length, we use task-specific training configurations rather than a single shared setting.

For localization, fixed-encoder learning freezes the protein encoder and trains a two-layer MLP on mean-pooled protein representations using Adam with a learning rate of $5e-5$, a global batch size of 128, and 100 epochs. Full-model tuning jointly optimizes the encoder and classification head using AdamW with encoder and head learning rates of $1e-5$ and $5e-4$, respectively, a global batch size of 16, and up to 100 epochs.

For fitness prediction, fixed-encoder learning trains the downstream regressor on frozen protein features using AdamW with a learning rate of $2e-4$. Full-model tuning progressively adapts the protein encoder using $1e-5$ learning rates. The training schedule, batch size, and regression-loss weights are selected separately for each fitness benchmark. In all cases, checkpoints are selected by validation Spearman's ρ.

For function annotation, we perform full-model tuning with task-specific configurations for EC, GO-BP, GO-MF, and GO-CC. EC uses an initial mean-pooled full-model predictor followed by an enhanced tuning stage. The GO branches similarly use task-specific full-model adaptation, with different layer ranges and optimization strengths for molecular function, biological process, and cellular component prediction. All function annotation models are selected by validation F_{max}. Detailed task-specific configurations are summarized in Table 5.

Evaluation Metrics. For localization, we report classification accuracy. For fitness prediction, we report Spearman's rank correlation ρ between the predicted and ground-truth fitness values,

$$\rho = \mathrm{corr}\big(\mathrm{rank}(y), \mathrm{rank}(\hat{y})\big). \tag{12}$$

For function annotation, we report pair-centric AUPR and protein-centric F_{max}. AUPR is computed by micro-averaging over all protein–function pairs. For F_{max}, let $\mathcal{T}_i$ denote the ground-truth functions of protein i and $\mathcal{P}_i(t)$ the functions predicted above threshold t. The precision and recall for

protein i are

$$p_i(t) = \frac{|\mathcal{P}_i(t) \cap \mathcal{T}_i|}{|\mathcal{P}_i(t)|}, \qquad r_i(t) = \frac{|\mathcal{P}_i(t) \cap \mathcal{T}_i|}{|\mathcal{T}_i|}. \tag{13}$$

We then compute

$$P(t) = \frac{1}{M(t)} \sum_{i:\,|\mathcal{P}_i(t)|>0} p_i(t), \qquad R(t) = \frac{1}{N} \sum_{i=1}^{N} r_i(t), \tag{14}$$

where N is the number of proteins and $M(t)$ is the number with at least one prediction at threshold t. Finally,

$$F_{\max} = \max_t \frac{2P(t)R(t)}{P(t) + R(t)}. \tag{15}$$

All checkpoints and score-processing choices are selected using the validation set before final test evaluation.