The Words of Proteins: Motif-Level Language Modeling for Interpretable Protein Generation

Anonymous Author(s)

Abstract

What are the fundamental units of protein sequences? Most protein language models treat amino acids as tokens, yet biological functions are not encoded at the single-residue level. Instead, they emerge from combinations of residues that form functional units, corresponding to conserved sequence motifs. Just like how modern language models work at the level of learned sub-word units instead of characters, we argue that explicitly modeling at the functional motif level provides both mechanistic insight into sequence-function relationships and interpretable control over protein generation. We demonstrate this framework on copper- and nickel-binding metalloproteins, where similar coordination chemistry is shared at the structure level yet how sequence controls binding specificity remains underexplored. We develop a three-step workflow: (1) construction of a dictionary of conserved subsequence motifs; (2) prediction of the next motif and inter-motif gaps; and (3) sequence infilling given the predicted anchor motifs. Amino acid enrichment analysis confirms that extracted motifs capture known metal-coordination chemistry (His, Cys, Met). Compared to random masking, motif-guided generation improves Conserved Domain Database annotation rates with greater target-family enrichment. Compared to finetuned ProtGPT2 that use BPE tokenization, our proposed functional motif-level approach achieves more specific hits on metalloprotein families and substantially reduces off-target annotations. Generation output directly mirrors dictionary composition, demonstrating that motif vocabularies provide explicit control over generation scope. AlphaFold 3 structure prediction with explicit metal ions confirms plausible coordination geometry, validating functional binding sites in generated sequences. Our results demonstrate that conserved sequence motifs carry functional information. Functional motif-level modeling enables interpretable, controllable protein generation, an important step toward compositional design of novel protein functions.

ACM Reference Format:
Anonymous Author(s). 2026. The Words of Proteins: Motif-Level Language Modeling for Interpretable Protein Generation. In . ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Protein language models (PLMs) have revolutionized computational biology by learning rich representations of protein sequences through self-supervised learning on large protein databases [18, 22]. Inspired by autoregressive and masked language models [6, 8], PLMs like ESM-2 [18] and ESM-3 [11] treat proteins as sequences of amino acid tokens and learn to predict masked or the next residues (the amino acids incorporated into proteins). These models have achieved remarkable success in protein sequence generation.

However, a fundamental disparity exists between how PLMs represent proteins and how Nature encodes biological functions. Biochemical research has established that protein function arises from specific arrangements of residues that form functional sites, binding pockets, and catalytic centers [5]. These functional units correspond to evolutionarily conserved sequence motifs that encode essential functions

This mismatch between representation granularity and functional encoding creates challenges for controllable generation of protein sequences with specific functions. When using masked PLMs like ESM-3 for protein design, users must specify which positions to mask. But without understanding of functional motifs, random or naive masking may disrupt critical functional regions.

Some PLMs have moved beyond single-residue tokens. This parallels the evolution of natural language processing, where modern language models use learned subword units such as Byte Pair Encoding [24] or WordPiece [23] rather than raw characters. For example, ProtGPT2 [10] employs BPE tokenization, requiring the model to implicitly learn functional motifs and evolutionary constraints from raw sequence data. However, BPE constructs its vocabulary through unsupervised merging of frequently co-occurring character pairs, yielding tokens that reflect corpus statistics rather than biological function.

Here we propose a different principle for vocabulary construction: grounding tokens in evolutionary signal rather than corpus statistics. We extract conserved subsequences from a target protein family and apply contrastive filtering against a background proteome, retaining motifs that are enriched within the family while discarding those that are ubiquitous. This procedure can be viewed as a form of implicit supervision from evolutionary selection, where motifs preserved within a family serve as positive examples of functional units and broadly distributed motifs act as negatives.

Beyond tokenization, prior generative protein design methods that leverage motifs typically use them as conditioning signals or structural anchors, often relying on structural information or learned embeddings without redefining the sequence representation itself [21, 27, 28]. Our approach instead redefines the modeling unit by treating conserved sequence motifs as discrete tokens and trains an autoregressive transformer to predict the next motif token without any structural input, directly embedding functional and evolutionary priors into the generative process.

This functional motif-level modeling provides several advantages for functional protein generation: (1) Interpretability: Since a

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

vocabulary of motifs is inherently more interpretable than a vocabulary of amino acids, this enables rational design of proteins with specific functions instead of working in a black box for researchers. (2) Controllability: By specifying motifs from which protein family to include, users can explicitly control the functional properties of generated proteins. The motif dictionary defines the space of possible functions. (3) Efficiency: Learning dependencies between functional units improves sample efficiency and achieves generation with functional precision.

We demonstrate our paradigm on metalloproteins, focusing on copper-binding (Cu(I) and Cu(II)) and nickel-binding (Ni(II)) protein families. Metalloproteins present an ideal test case for several reasons. First, metal coordination is a well-characterized function with known sequence determinants: residues such as histidine, cysteine, and methionine are established metal-coordinating amino acids [13]. This provides ground truth for validating whether our extracted motifs capture biologically relevant features. Second, despite sharing similar coordination chemistry and binding pocket geometry at the structural level, how different metalloprotein subfamilies employ distinct sequence patterns to determine metal specificity remains elusive.

We develop a three-component workflow for motif-level protein generation: (a) family-specific motif dictionary construction, where we extract conserved subsequences from target protein families and filter against a background proteome to retain only function-specific motifs; (b) a motif-level language model, where we train a transformer to predict the next motif and inter-motif gap length, learning the grammar of how functional units combine within proteins; and (c) scaffold generation and infilling, where the transformer generates scaffolds with motifs as anchors and ESM-3 fills the intervening gaps while preserving functional constraints.

Analysis demonstrates that our motif dictionary on metalloproteins is enriched in known metal-coordinating residues (His, Cys, Met), confirming that unsupervised extraction recovers genuine functional features. In generation experiments, motif-guided masking achieves higher functional recovery than random masking, and our functional motif-level generation outperforms BPE tokenization model (ProtGPT2) with more target-family hits and fewer off-target annotations. Notably, the transformer's output distribution mirrors dictionary composition, demonstrating the predictable and interpretable control of generation. AlphaFold 3 [1] predicted structures confirm plausible metal coordination geometry (ipTM 0.91–0.93) in generated sequences.

With this paper, we make the following contributions:

1. Conceptual reframing: we propose modeling proteins at the functional motif ("meaningful word") level. Like BPE in NLP, our work modifies the tokenization scheme, not architecture or training objective. The distinction lies in the vocabulary: BPE optimizes for corpus compression; we optimize for functional specificity via evolutionary conservationan.
2. Practical workflow: We develop an end-to-end pipeline for motif-level protein generation, integrating dictionary construction, transformer-based scaffold generation, and PLM-based infilling.
3. Interpretability demonstration: We demonstrate on metalloproteins that this approach provides a form of interpretability absent in end-to-end PLMs: dictionary composition directly predicts generation output, giving users explicit, inspectable control over functional scope.

2 Related Work

Protein Language Models. The application of language modeling techniques to protein sequences has yielded remarkable advances in computational biology. Early work demonstrated that recurrent neural networks could learn meaningful representations from protein sequences [2]. The introduction of transformer-based models dramatically improved performance: Evolutionary Scale Modeling (ESM) showed that unsupervised learning on protein sequences produces representations that capture structural and functional properties [22]. ESM-2 scaled this approach to 15 billion parameters, achieving state-of-the-art performance on structure prediction tasks [18]. Most recently, ESM-3 demonstrated that a single model trained on sequence, structure, and function can generate novel functional proteins [11].

For protein generation, autoregressive models have shown particular promise. ProtGPT2 adapts the GPT-2 architecture to protein sequences and can generate novel proteins that fold into realistic structures [10]. ProGen uses conditional language modeling to generate proteins with desired properties [20]. These models operate at the amino acid level, treating each residue as a token.

Our work differs fundamentally in the granularity of modeling. Rather than predicting individual amino acids, we predict conserved motifs, which marks a shift from "character-level" to "word-level" protein language modeling. This enables explicit representation of functional units that amino acid-level models must learn implicitly.

Tokenization in Natural Language Processing. The question of what constitutes the right "unit" for sequence modeling has been extensively studied in NLP. Character-level models offer open vocabularies but struggle to capture long-range dependencies [16]. Word-level models capture meaning more directly but face out-of-vocabulary challenges.

Modern language models have converged on subword tokenization as a practical middle ground. Byte Pair Encoding (BPE) iteratively merges frequent character pairs to build a vocabulary of variable-length subword units [24]. WordPiece uses a similar approach optimized for likelihood [23]. SentencePiece provides a language-agnostic implementation [17]. These methods are now standard in models from BERT to GPT-4. The success of subword tokenization suggests that the granularity of representation matters for sequence modeling. However, BPE and WordPiece derive tokens from corpus statistics—frequent co-occurrences—without reference to meaning. In contrast, our protein motifs are defined by evolutionary conservation, which directly reflects functional importance. This grounds our tokenization in biological signal rather than statistical patterns.

Recent work has begun exploring alternative tokenization strategies for proteins. RITA examined the effect of BPE on protein language models [12]. However, to our knowledge, no prior work has proposed using evolutionarily conserved motifs as the fundamental units for protein language modeling.

Controllable and Constrained Generation. Controlling the output of language models is an active research area. Approaches fall

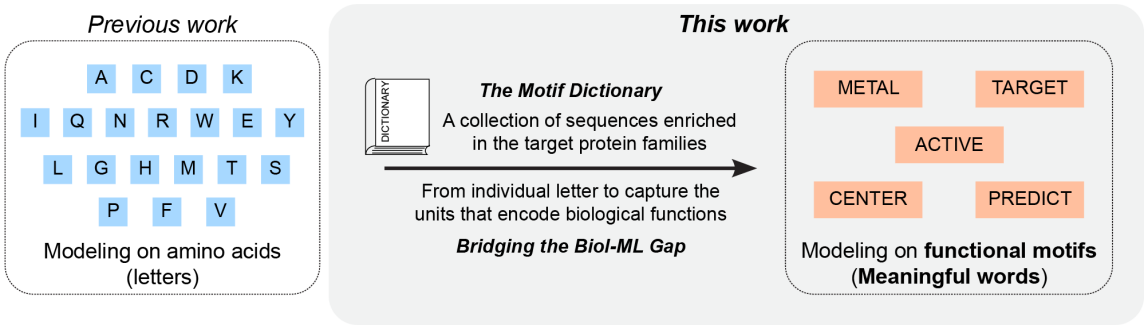

Figure 1: Overview of proposed functional motif-level language modeling for functional protein generation. Words shown as conceptual analogy; actual motifs are conserved amino acid sequences.

into several categories. The first is to condition the output on the input, but there is no guarantee that the learned model may respect the input constraints [15]. For protein generation, [20] explores conditioning on functional labels or structural features. The second type of methods, like NeuroLogic [19], explicitly enforce constraints on the model output. ILM [9] meets constraints by performing infilling of partial sequences. Finally, one may finetune models on curated data that satisfy the constraints in the hope that the models learn the desired behaviors [14].

Our approach combines elements of constrained decoding and conditioning. By specifying which motifs must appear in the generated sequence, we provide hard constraints on functional content. The motif dictionary serves as a form of conditioning that explicitly defines the functional scope of generation. Unlike soft prompting approaches, our constraints are interpretable and directly tied to biological function.

3 Method and Experimental Setup

3.1 Data curation

Metalloproteins. Protein sequences were retrieved from the RCSB Protein Data Bank (PDB), which archives experimentally determined protein structures [4]. Using the RCSB PDB API, we queried structures annotated with Cu(I), Cu(II), or Ni(II) ligands and extracted the corresponding protein chain sequences from the FASTA endpoint. To improve dataset quality and minimize length-driven bias in downstream analyses, we removed extreme-length chains below the 10th percentile and above the 90th percentile of the length distribution, filtering away truncated fragments as well as unusually long multi-domain chains like fusion or chimeric constructs that are common in structural datasets. We further removed nucleotide-like sequences as a defensive quality-control step and deduplicated sequences by exact amino-acid identity to avoid over-representing repeated chains across related structures; in addition to within-label deduplication, exact duplicates occurring across the Cu(I), Cu(II), and Ni(II) sets were removed to ensure that each unique sequence was counted only once across metalloprotein classes. After filtering, the metalloprotein dataset comprised 193 Cu(I), 1,906 Cu(II), and 1,107 Ni(II) sequences.

Background Proteome. A non-metal background proteome was assembled from the RCSB Protein Data Bank using the same RCSB PDB API. All protein polymer entities were obtained from RCSB, and subsequently filtered to exclude structures annotated with common metal ions and metal-associated cofactors based on non-polymer component identifiers. Specifically, entries containing any of the following components were excluded: ZN, FE, FE2, FE3, CU, CU1, CUA, NI, CO, MN, MO, W, V, FES, SF4, F3S, HEM, HEA, HEB, HEC, HEO, CUZ, and CUB. Protein chain sequences were then retrieved from the FASTA endpoint for each retained entry. Except for the length-based filtering, the same sequence quality-control and deduplication procedures described above were applied to the background proteome prior to motif mining, and the resulting background proteome has a total of 145,318 unique sequences.

3.2 Dictionary creation

Motif Mining. We exhaustively enumerated all overlapping length-5 amino-acid substrings (5-grams) u_5 from each sequence $s \in \mathcal{S}$, yielding the set of all 5-grams observed in the metalloprotein sequences. We start from 5-grams because prior work suggests that this length captures informative local protein sequence patterns while remaining robust to minor variation [25].

For any substring u, we define its support set $S(u)$ as the set of sequences in which u occurs at least once. We extended each u_k by one residue to the left or right to obtain u_{k+1}, but only when $S(u_{k+1}) = S(u_k)$. When no such extension is possible, we save the resulting longest motif m associated with that support set. To remain inclusive while still requiring conservation across sequences, we retained only motifs with $|S(m)| \geq 2$ for downstream analyses. We also ensured all $(m, S(m))$ pairs were unique.

After extension, we removed redundant motifs. For each fixed support set, we removed any motif m that is a strict contiguous substring of a longer motif m' with the same support set. Because substring pruning is restricted to motifs sharing the same support set, motifs supported by different sequence sets are not merged or pruned, allowing a shorter motif with broader support and a longer motif with narrower support to coexist. The full algorithm is detailed in Appendix A.

Motifs Filtering. Motif mining on the metalloprotein datasets yielded 57,699 unique motifs initially. To enrich for motifs that are more likely to be functionally related to metal binding, we filtered these candidates against a non-metal background proteome.

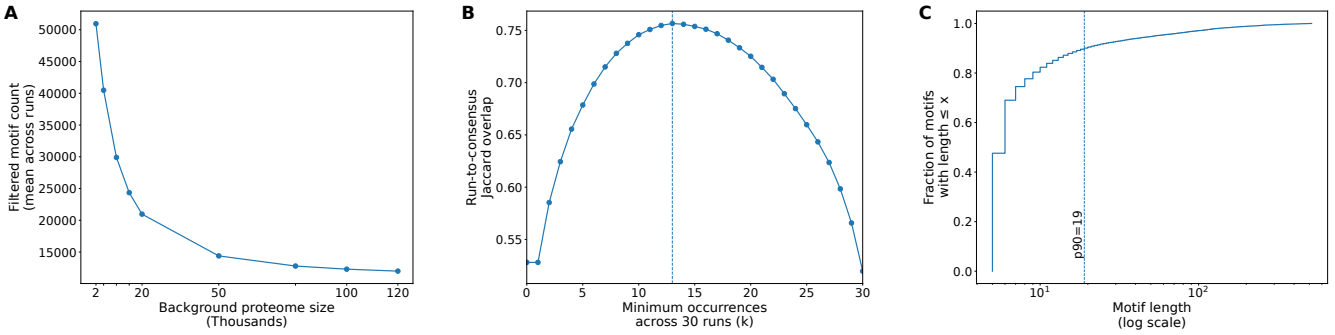

Figure 2: Selection of parameters for constructing dictionary. (A) Number of conserved motifs decreases with increasing background proteome size. Background proteome size of 20,000 was chosen as the appropriate background proteome size. (B) Conserved motifs with more than 13 times in occurrence is shown to be the most stable after repeated downsampling of background proteome. (C) To prevent extremely long motifs, retained motifs longer than the 90th percentile in length were discarded.

Specifically, we mined motifs from the background proteome using the same pipeline and removed any motifs that also appeared in the background set, retaining only motifs that were present in metalloproteins but absent from the non-metal background.

Using the entire background proteome (n=145,318) for filtering can be overly stringent, as it may discard motifs that are broadly conserved yet still contribute to metalloprotein function. We therefore varied the background size by repeated downsampling across a range of background proteome sizes (2,000; 5,000; 10,000; 15,000; 20,000; 50,000; 80,000; 100,000; 120,000 sequences). We sampled each size 30 times independently and quantified the number of motifs retained after filtering. A background size of 20,000 was selected because it lay at the elbow of the motif retention curve, representing a practical balance between removing ubiquitous, non-specific motifs and preserving a sufficiently rich motif set for downstream modelling (Fig. 2A).

Because each subsample changes the underlying set of background sequences, the mined background motifs are not identical across runs. To obtain a robust background motif dictionary, we tracked how often each motif was recovered across the 30 independent sampling of 20,000 background sequences, and used a minimum-occurrence threshold to retain recurrent motifs while suppressing subsample-specific noise. We selected k=13 as the minimum occurrence threshold, as this value maximised run-to-run concordance of the resulting consensus motif set, indicating that the retained motifs were consistently recovered across independent background subsamples (Fig. 2B).

Finally, we inspected the length distribution of the retained motifs and observed a pronounced inflation in the upper tail, with the longest motifs disproportionately concentrated in the top 10% of the distribution. To avoid admitting unwieldy, sequence-like motifs that may reflect multi-domain or chimeric constructs rather than localisable functional patterns, we removed these extreme-length motifs and retained only motifs which are equal or less than the 90th percentile in length (19 residues) (Fig. 2C). The number of motifs in our final dictionary is 18,954.

Conceptually, this set of procedures mirrors classical vocabulary curation in natural language processing: the background-proteome filter acts like stop-word removal (excluding high-frequency, non-discriminative patterns), the minimum-occurrence criterion acts like a document-frequency threshold (retaining tokens that recur reliably across resamples), and the length filter functions as a safeguard against degenerate "tokens" that effectively become whole sentences rather than reusable subunits.

3.3 Validating utility of dictionary

Amino acid enrichment. We assessed whether conserved motifs were enriched for specific amino-acid residues relative to motifs removed during the filtering steps described earlier. For each of the 20 canonical amino acids, residue counts were contrasted between the conserved and removed sets. Fisher's exact test was used to identify residues significantly overrepresented in conserved motifs.

Conserved Domain Recovery. Using the curated motif dictionary, we generated motif-guided masked scaffolds of each metalloprotein sequence by masking all non-conserved regions and preserving only residues belonging to the identified motifs. Because sequences differed in the fraction of residues covered by motifs, we stratified motif retained sequences into five bins based on masking ratios: 0–0.2, 0.2–0.4, 0.4–0.6, 0.6–0.8, and 0.8–1.0, where 0 represents the original unmasked sequence and 1.0 means a fully masked sequence.

To benchmark functional signal retention, we constructed a matched random-masking control for each sequence. Specifically, we applied random masks at the same overall mask ratio as the motif-guided masked sequence, while maximizing overlap between random masks and motif regions to create a conservative control.

Masked regions in both masking strategies were infilled using zero-shot ESM3 generation [11] to yield full-length sequences. We repeated the random masking and zero-shot generation 30 times to ensure the downstream analyses were stable across stochastic runs.

We obtained the infilled sequences' functional annotations using NCBI's Conserved Domain Database (CDD) via CD-Search [26].

The original unmasked sequences were processed through the same CDD pipeline to provide a reference set of annotations.

We quantified annotation recovery by testing whether the CDD annotation assigned to an infilled sequence matched that of its original unmasked sequence. Let $a_{\text{orig}}(s)$ denote the CDD domain annotation returned for the original sequence s, and let $a_{\text{mm}}(s)$ and $a_{\text{rm}}(s)$ denote the corresponding annotations for the motif-masked–infilled and random-masked–infilled versions of s, respectively. For each sequence, we define a boolean indicator of annotation recovery:

$$\mathbb{I}_{\text{mm}}(s) = \mathbf{1}\{a_{\text{mm}}(s) = a_{\text{orig}}(s)\}, \qquad \mathbb{I}_{\text{rm}}(s) = \mathbf{1}\{a_{\text{rm}}(s) = a_{\text{orig}}(s)\}.$$

Recovery rates were then computed within each masking-ratio bin b as the proportion of sequences whose top annotation is recovered:

$$R_{\text{mm}}(b) = \frac{1}{|\mathcal{S}_b|} \sum_{s \in \mathcal{S}_b} \mathbb{I}_{\text{mm}}(s), \qquad R_{\text{rm}}(b) = \frac{1}{|\mathcal{S}_b|} \sum_{s \in \mathcal{S}_b} \mathbb{I}_{\text{rm}}(s),$$

where $\mathcal{S}_b$ denotes the set of sequences in bin b. We compared $R_{\text{mm}}(b)$ and $R_{\text{rm}}(b)$ to assess whether motif-guided masking preserves functional signal better than the matched random-masking control.

3.4 Model Training

Data preparation. For each subtype of metalloproteins, motif-retained sequences were split into training and validation sets (90% and 10%, respectively). These sequences can be tokenized into: motif tokens (`M: <motif>`), gap tokens (the non-conserved regions being removed) (`G:<gap_length>`), and overlap tokens (`G:<overlap_length>`) to represent cases where motifs overlap into a single contiguous region. Each token stream was prefixed with a `<SUBTYPE>` token to provide subtype context during training. To avoid out-of-vocabulary tokens at validation time, the validation set was constrained such that all motifs appearing in validation also appeared in the corresponding training split. Token streams were then mapped to integer IDs and standardized to a fixed context length via truncation and right-padding; padding positions were masked and excluded from loss computation during training.

Training. We trained subtype-specific autoregressive Transformer language models over motif–gap token streams using a GPT-2–style decoder-only architecture. Models were optimised with a standard maximum-likelihood next-token objective, minimising cross-entropy between the predicted token distribution and the ground-truth next token at each position. We used AdamW with gradient clipping to improve optimisation stability. Model performance was monitored on a held-out validation set evaluated once per epoch; we report validation loss, and retain the best-performing checkpoint according to the validation loss.

For each subtype, we selected the best hyperparameter configuration by lowest validation loss from our grid search across 50 epochs. The selected configurations were: Cu(I) (learning rate 3×10^{-4}, batch size 16, context length 256, embedding dimension 512, 6 transformer layers, 8 attention heads), Cu(II) (learning rate 1×10^{-4}, batch size 16, context length 384, embedding dimension 512, 6 layers, 8 heads), and Ni(II) (learning rate 2×10^{-4}, batch size 16, context length 256, embedding dimension 384, 6 layers, 8 heads). Further details are reported in Appendix B.

3.5 Protein Sequence Generation

Constrained token-stream generation. Given a subtype prompt (`<SUB=...> <BOS>`), we sample tokens step-wise using nucleus (top-p) sampling with temperature under a hard grammar that enforces alternation

$$\texttt{M:}* \rightarrow \texttt{G:}* \rightarrow \texttt{M:}* \rightarrow \cdots$$

Termination permits `<EOS>` only after a minimum number of motifs, with a soft `<EOS>` bias beyond a chosen motif count and a hard maximum motif cap.

Gap-fraction steering. At a gap step (immediately after a motif), we track the current gap-fraction of the generated stream $\hat{f}_{\text{gap}}(t)$, and compare it against a user-specified gap-fraction $f^\star$

$$L_{\text{motif}}(t) = \sum_{i \le t : z_i \in \mathcal{M}} \ell(z_i), \qquad L_{\text{gap}}(t) = \sum_{i \le t : z_i \in \mathcal{G}} g(z_i),$$

$$\hat{f}_{\text{gap}}(t) = \frac{L_{\text{gap}}(t)}{L_{\text{gap}}(t) + L_{\text{motif}}(t)}, \qquad e_t = f^\star - \hat{f}_{\text{gap}}(t).$$

Among the possible gap lengths $g \in \mathcal{G}_t$, we sample

$$p(g_t = g \mid z_{<t}) \propto p_\theta(g \mid z_{<t}) \exp(\beta e_t g),$$

with $\beta > 0$ (zero-length separators, including overlaps, use $g = 0$). This implements an online feedback controller: when $e_t > 0$ (below target) longer gaps are upweighted, and when $e_t < 0$ (above target) shorter gaps are favoured, steering $\hat{f}_{\text{gap}}(t)$ toward $f^\star$ throughout decoding.

Hard overlap enforcement. When an overlap token of length k is selected after motif m_{t-1}, the next motif is restricted to

$$\mathcal{M}_t = \{m \in \mathcal{M} : \text{prefix}_k(m) = \text{suffix}_k(m_{t-1})\},$$
$$p(m_t = m \mid z_{<t}) \propto p_\theta(m \mid z_{<t})\, \mathbb{I}[m \in \mathcal{M}_t].$$

This ensures that overlap-joined motifs are string-consistent and can be merged into a contiguous sequence.

Detokenization and infilling. The generated token stream was converted into an amino-acid scaffold by concatenating motif strings and inserting placeholder spans for each unfilled gap token. Overlap tokens were realized by merging adjacent motifs with the specified shared substring. This produced a partially specified sequence containing fixed motifs and undefined regions, which were subsequently infilled using a zero-shot ESM3 generation [11] to yield full-length sequences.

3.6 Generated Protein Sequence Evaluation

We evaluated our generated sequences on functional annotation and structural validation by AlphaFold 3.

ProtGPT2 generation comparison. We fine-tuned ProtGPT2 [10] independently for each metalloprotein family (Cu(I), Cu(II), Ni(II)). We used the default hyperparameters suggested by the authors of ProtGPT2 for finetuning, including a context window of 256 tokens and a learning rate of 10^{-3}. We used the same 90/10 train/validation split as for our motif-tranformer (Section 3.4), with the validation file used to monitor language-model fit during fine-tuning, and no hyperparameters were tuned against the final evaluation metric. This produces three metalloprotein-family-specific sequence generators for comparison against our motif-transformer.

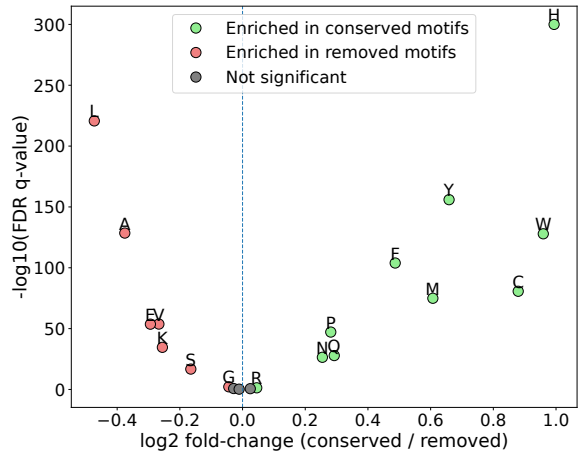

Figure 3: Motif-level amino acid enrichment analysis

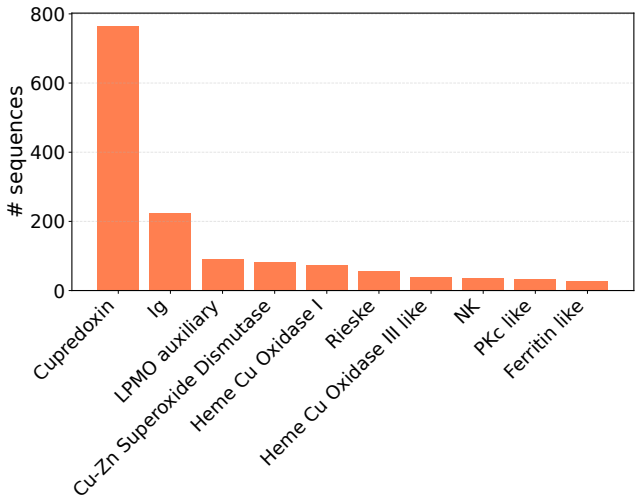

Figure 4: Functional annotations on our motif dictionary against Conserved Domain Database

For comparison with our motif-transformer, we sampled sequences from the fine-tuned ProtGPT2 models and evaluated their functional annotations produced from the pipeline described earlier (CDD annotation via CD-search).

Structure Prediction and Visualization. Metal ions were added using their Chemical Component Dictionary codes to simulate binding with the generated sequences in Alphafold3. All other model parameters were maintained as per their default settings [1]. Evaluation was done using the interface predicted template modelling (ipTM) score to quantify the accuracy of predicted relative positions between the metal ion and the protein. ipTM values exceeding 0.80 were taken to be of high confidence, suggesting good interaction between the metal and the protein. Values within the range of 0.60 and 0.80 required more scrutiny whereas those below 0.60 were disregarded. Further validation was also done by visual inspection in PyMOL [7] to confirm the coordination centers. All structures shown were prepared by PyMOL.

4 Results and Discussion

4.1 Validation of Dictionary Functional Content

We validated that our motif dictionary captures biologically meaningful functional information through two complementary analyses at different levels of granularity.

4.1.1 Residue-Level Validation. We envision that if our motif-based tokenization captures the biological functions, the residues that support metal-coordination chemistry should show higher frequencies, i.e., be enriched. Therefore, we performed amino acid enrichment analysis on our motif dictionary against those filtered out as negatives. Fig. 3 shows a volcano plot of log2 fold-change against statistical significance (-log10 FDR q-value). The amino acids most enriched in conserved motifs—histidine (H), cysteine (C), methionine (M), tyrosine (Y), and tryptophan (W)—are precisely those known from biochemistry to coordinate metal ions [13]. This is notable because our extraction method has no knowledge of metal chemistry and operates purely at a statistical level. Conversely, amino acids enriched in removed motifs, such as leucine (L), alanine (A), valine (V), are generic hydrophobic residues common across all proteins. This analysis serves as ground-truth validation: our unsupervised motif extraction independently recovers established domain knowledge about metal-binding residues.

4.1.2 Domain-Level Validation: Conserved Domain Annotation. We next performed the functional annotation of our motif dictionary against the Conserved Domain Database (CDD) to evaluate: 1) whether our motif dictionary captures the metal-binding biological functions, and 2) the distribution of functional scopes (i.e., subfamilies of metalloproteins) encoded by our motif dictionary. Fig.4 shows the distribution of CDD annotations across dictionary motifs. The dictionary is dominated by Cupredoxin motifs (~770 sequences), the canonical copper-binding family. The second largest category, Immunoglobulin-like domains (~230), initially appears surprising—Ig folds are typically associated with immune function rather than metal binding. However, literature confirms that certain immunoglobulins can bind copper with high affinity [3]. The presence of these non-canonical metal binders demonstrates that our unsupervised extraction captures functional relationships beyond obvious metalloprotein families. The remaining entries span established metalloprotein families: LPMO auxiliary domains (~90), Cu-Zn Superoxide Dismutase (~85), Heme Cu Oxidases, and Rieske iron-sulfur proteins.

Together, these analyses validate that the dictionary encodes metal-binding function at both the residue level (coordination chemistry) and domain level (protein family membership). Importantly, the method recovers known biochemistry and extends it to non-obvious cases, all without explicit supervision. This provides biological grounding for our motif-level language modeling approach: our motif vocabulary corresponds to genuine functional units.

4.2 Generation Results

We evaluate generation quality through three comparisons: (1) random versus motif-guided masking, (2) function-encoded motif

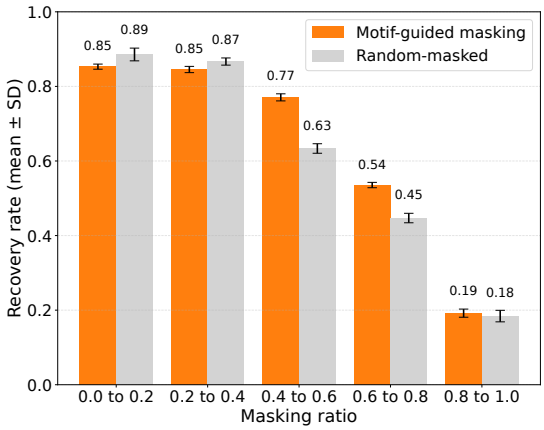

Figure 5: CDD annotation recovery rates for random and motif-guided masking, with the mean value annotated.

tokenization versus BPE tokenization, and (3) alignment between dictionary composition and generation output.

4.2.1 Random versus Motif-guided Masking. To test whether our function-encoded motif vocabulary ("words") can truly improve generation of functional protein sequences ("sentences"), we compared the functional annotation on ESM3 generated sequences with motif-guided masked scaffolds (keeping the conserved motifs in the original sequence and removing the non-conserved sequences as gaps) versus random-masked scaffolds (masking arbitrary positions at matched ratios). The rationale is that not much difference would be expected if ESM3 (model that uses residue-level tokenization) learned all functional features.

Fig.5 shows functional annotation recovery rates (i.e, the ratio of sequences that can be funtionally annotated) across different masking ratios. When masking ratio is very low (0-0.2) or very high (0.8-1.0), both methods achieve similar recovery (~90% at low ratio; ~20% at high ratio). However, at moderate masking ratios (0.4–0.8), motif-guided masking substantially outperforms random masking. Furthermore, recovery rate on random masking dropped significantly when the masking ratio is higher than 0.4, while motif-guided masking remains optimal performance at 0.4-0.6 ratio.

This result establishes that motifs encode functional information that ESM3 alone does not capture, preserving them is necessary for functional generation. Destroying conserved sequence motifs by random masking results in impaired functional regions, leading to sequences that ESM3 can fill in but no longer encode recognizable protein functional families. In contrast, motif-guided masking preserves functional anchors, enabling ESM3 to generate sequences that maintain biological functions.

4.2.2 Function-encoded Motif Tokenization versus BPE Tokenization. Having established that motifs encode functional information, we evaluate end-to-end generation quality: our full pipeline (motif transformer + ESM3 infilling) versus ProtGPT2 (BPE-based autoregressive generation). ProtGPT2 was fine-tuned on the same metalloprotein training set as our motif transformer. For each metal type (Cu(I), Cu(II), Ni(II)), we generated 100 sequences per method

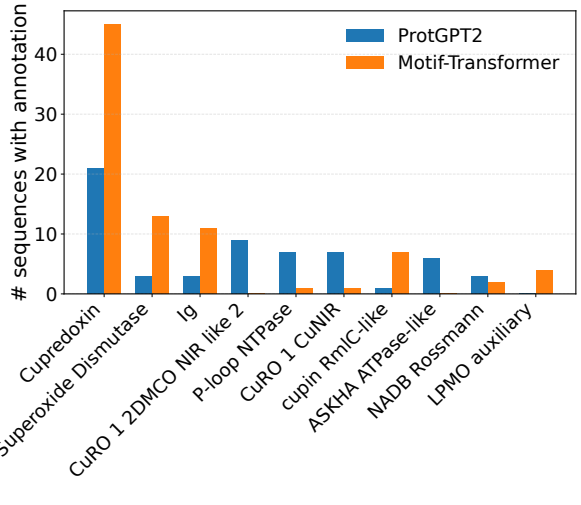

Figure 6: CDD annotations of protein sequences generated by ProtGPT2 (BPE tokenization) vs. our motif-transformer.

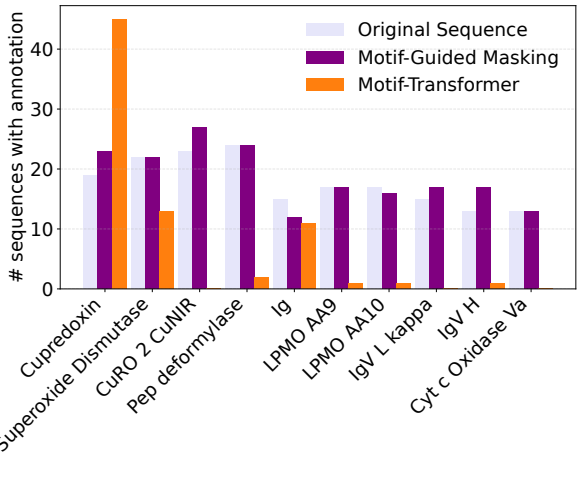

Figure 7: CDD annotations of protein sequences generated by deterministic reconstruction (motif-guided masking) vs. learned selection (our motif-transformer).

(300 total per method) and performed functional annotation against CDD. The key difference is how sequence units are defined: we use conserved motifs grounded in evolutionary signal while ProtGPT2 uses BPE tokens derived from corpus statistics.

Fig.6 shows pooled results across all metal types. Motif-level generation produces substantially more hits on target metalloprotein families: 114% more Cupredoxin (45 vs. 21), 4× more Cu-Zn Superoxide Dismutase (13 vs. 3), and exclusive hits to families such as Immunoglobulin-like domains and LPMO auxiliary. Critically, motif-level generation nearly eliminates off-target annotations: P-loop NTPase, which is an ATP/GTP binding domain unrelated to

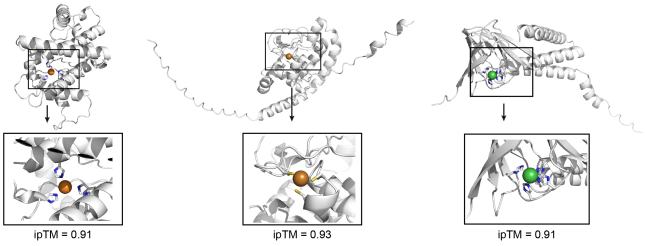

Figure 8: Examples of generated protein structures with metal ions. ipTM values suggest confident metal-binding.

metal coordination, drops from 7 hits to 1; while ASKHA ATPase-like domains are eliminated entirely. The results stratified by metal type are reported in Appendix C, showing consistent patterns across Cu(I), Cu(II), and Ni(II) with the strongest advantage observed for Cu(I)-binding proteins.

These results demonstrate that biologically-grounded tokenization outperforms statistically-derived tokenization for functional protein generation. Both approaches were trained on identical data, yet motif-level generation produces substantially greater target-family enrichment and fewer off-target hits. The difference lies in what each vocabulary encodes: our motifs explicitly capture metal-binding sequence patterns, while BPE tokens that are optimized for compression must learn these patterns implicitly, and appear to conflate metal-binding with generic nucleotide-binding patterns.

4.2.3 Rule-based vs. Motif-Transformer. We next asked what the transformer has learned beyond simple motif preservation. We examined the generation behavior by comparing two scaffold construction strategies: (1) deterministic reconstruction, where motifs are copied from input sequences at their original positions (rule-based), and (2) motif-transformer, a generative sampling, where the transformer samples motifs from a learned distribution. Both scaffolds are infilled by ESM3.(Fig.7).

Rule-based generation closely mirrors the original sequence distribution: Cu-Zn Superoxide Dismutase, CuRO, and Pep deformylase appear at similar frequencies in both original and reconstructed outputs. This is expected—preserving motifs at their natural positions faithfully reconstructs the input family composition. The transformer produces a strikingly different distribution. Cupredoxin, Immunoglobulin-like domains, and LPMO families are amplified, while Cu-Zn SOD, CuRO, and Pep deformylase are suppressed. This shifted distribution, however, mirrors the dictionary composition (Fig.4), not the input sequences: Cupredoxin dominates the dictionary and dominates transformer output; Ig and LPMO are well-represented in the dictionary and amplified in generation; CuRO motifs are rare in the dictionary and rarely generated.

This reveals what the transformer has learned: the statistical structure of the motif vocabulary. The dictionary-to-generation correspondence provides a form of interpretability absent in black-box models: users can inspect the vocabulary to predict generation behavior, and curate the vocabulary to control functional scope. This represents a qualitatively different form of controllability than prompt engineering or fine-tuning: control is achieved through vocabulary curation rather than model modification.

4.3 Structural Validation

Sequence-based metrics (CDD annotation) confirm that generated proteins match known family signatures, but do not directly assess whether the proteins would fold correctly and coordinate metals. To validate structural plausibility, we used AlphaFold 3 to predict structures of generated sequences with explicit metal ions. Sequences used are reported in Appendix D.

Fig.8 shows three representative examples spanning different metal types and protein architectures. Each structure was predicted with the appropriate metal ion (copper or nickel) included as an input ligand. The interface predicted Template Modeling score (ipTM) measures confidence in the protein-metal interface, where values above 0.8 indicate high-confidence predictions. All three examples achieve ipTM scores above 0.9 (0.91, 0.93, and 0.91), indicating that AlphaFold 3 confidently predicts stable protein-metal interfaces. The zoomed views reveal chemically plausible coordination geometry: histidine residues (blue) and cysteine residues (yellow) position their nitrogen and sulfur atoms toward the metal centers, consistent with established metalloprotein coordination chemistry.

These results demonstrate that our generated sequences encode structural information sufficient for AlphaFold 3 to predict confident metal-binding sites with correct coordination chemistry. While computational structure prediction does not guarantee experimental function, the high ipTM scores suggest that our motif-level generation framework produces sequences with realistic structures.

5 Conclusion

We have proposed a shift in how proteins are modeled for generation: from statistically-derived tokens to biologically-grounded motifs. A change in vocabulary construction alone, while keeping architecture and training objective standard, fundamentally shapes generation outcomes. Biological knowledge is incorporated at the tokenization level, where evolutionary conservation is used to define higher-level units, reducing the complexity of the learning problem. The interpretability of our approach, where dictionary composition directly predicts generation output, enables rational design strategies that black-box PLMs cannot support. By curating motif dictionaries for target functions, researchers can generate candidate proteins enriched in desired properties prior to wet-lab validation. This paradigm opens avenues for future research, including combining vocabularies from multiple families for compositional design. More broadly, our work demonstrates that domain knowledge can be embedded into ML pipelines through vocabulary construction, a principle we anticipate extends beyond proteins to other scientific domains where expert knowledge can define meaningful sequence units.

6 Limitations and Ethical Considerations

While our results on the metalloproteins are promising, generalization to other protein families remains to be validated. Protein generation tools could theoretically be misused to design harmful proteins, but our method is constrained to generate proteins similar to those already in training data. We will release our code, motif dictionaries, and generated sequences to enable independent verification and extension of our work.

7 GenAI Disclosure

Generative AI has been used to improve spelling, grammar, punctuation and clarity of this manuscript. All text, figures, tables and code are produced by the authors.

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. 2024. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature* 630, 8016 (2024), 493–500.

[2] Ethan C Alley, Grigory Khimulya, Surojit Biswas, Mohammed AlQuraishi, and George M Church. 2019. Unified rational protein engineering with sequence-based deep representation learning. *Nature methods* 16, 12 (2019), 1315–1322.

[3] Bennie L Baker and DONALD E Hultquist. 1978. A copper-binding immunoglobulin from a myeloma patient. Purification, identification, and physical characterization. *Journal of Biological Chemistry* 253, 4 (1978), 1195–1200.

[4] H. M. Berman. 2000. The Protein Data Bank. *Nucleic Acids Research* 28, 1 (Jan 2000), 235–242. doi:10.1093/nar/28.1.235

[5] Peer Bork and Eugene V Koonin. 1996. Protein sequence motifs. *Current opinion in structural biology* 6, 3 (1996), 366–376.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[7] Warren L DeLano et al. 2002. Pymol: An open-source molecular graphics tool. *CCP4 Newsl. protein crystallogr* 40, 1 (2002), 82–92.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[9] Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. *arXiv preprint arXiv:2005.05339* (2020).

[10] Noelia Ferruz, Steffen Schmidt, and Birte Höcker. 2022. ProtGPT2 is a deep unsupervised language model for protein design. *Nature communications* 13, 1 (2022), 4348.

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

[12] Daniel Hesslow, Niccoló Zanichelli, Pascal Notin, Iacopo Poli, and Debora Marks. 2022. Rita: a study on scaling up generative protein sequence models. *arXiv preprint arXiv:2205.05789* (2022).

[13] Richard H Holm, Pierre Kennepohl, and Edward I Solomon. 1996. Structural and functional aspects of metal sites in biology. *Chemical reviews* 96, 7 (1996), 2239–2314.

[14] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*. PMLR, 2790–2799.

[15] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858* (2019).

[16] Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. 2016. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 30.

[17] Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226* (2018).

[18] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, et al. 2022. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *BioRxiv* 2022 (2022), 500902.

[19] Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Neurologic decoding:(un) supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human language technologies*. 4288–4299.

[20] Ali Madani, Ben Krause, Eric R Greene, Subu Subramanian, Benjamin P Mohr, James M Holton, Jose Luis Olmos Jr, Caiming Xiong, Zachary Z Sun, Richard Socher, et al. 2023. Large language models generate functional protein sequences across diverse families. *Nature biotechnology* 41, 8 (2023), 1099–1106.

[21] Kevin Michalewicz, Chen Jin, Philip Alexander Teare, Tom Diethe, Mauricio Barahona, Barbara Bravi, and Asher Mullokandov. 2025. Protein generation with embedding learning for motif diversification. *arXiv preprint arXiv:2510.18790* (2025).

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

[23] Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 5149–5152.

[24] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*. 1715–1725.

[25] Taha ValizadehAslani, Zhengqiao Zhao, Bahrad A Sokhansanj, and Gail Rosen. 2020. Amino Acid k-mer Feature Extraction for Quantitative Antimicrobial Resistance (AMR) Prediction by Machine Learning and Model Interpretation for Biological Insights. *Biology* 9, 11 (Oct 2020), 365–365. doi:10.3390/biology9110365

[26] Jiyao Wang, Farideh Chitsaz, Myra K Derbyshire, Noreen R Gonzales, Marc Gwadz, Shennan Lu, Gabriele H Marchler, James Song, Narmada Thanki, Roxanne A Yamashita, Mingzhang Yang, Dachuan Zhang, Chanjuan Zheng, Christopher J Lanczycki, and Aron Marchler-Bauer. 2022. The conserved domain database in 2023. *Nucleic Acids Res.* 51, D1 (Dec 2022), D384–D388. doi:10.1093/nar/gkac1096

[27] Bo Zhang, Kexin Liu, Zhuoqi Zheng, Junjie Zhu, Zhengxin Li, Yunfeiyang Liu, Junxi Mu, Ting Wei, and Hai-Feng Chen. 2025. Protein language model supervised motif-scaffolding design with GPDL. *International Journal of Biological Macromolecules* (2025), 148441.

[28] Zhuoqi Zheng, Bo Zhang, Kieran Didi, Kevin K Yang, Jason Yim, Joseph L Watson, Hai-Feng Chen, and Brian L Trippe. 2025. MotifBench: A standardized protein design benchmark for motif-scaffolding problems. *arXiv preprint arXiv:2502.12479* (2025).

Appendix

A Motif Mining Algorithm

Algorithm 1: Conserved motif mining by support-invariant motif extension and support-conditioned pruning

Input: Sequences $\{x_s\}_{s\in\mathcal{S}}$; motif length k (default 5); minimum support τ (default 2).
Output: Motifs $\mathcal{M}$ with support sets $S(m)$.

Support set.;
For any substring u, define

$$S(u) = \{\, s \in \mathcal{S} : \exists p,\ x_s[p : p + |u|] = u \,\}.$$

1) Enumerate k-mer seeds.;
Enumerate all overlapping k-mers u by sliding a window over each x_s, and record $S(u)$.;
Let $\mathcal{U} \leftarrow \{u :\ u$ is a k-mer observed in at least one $x_s\}$.;

2) Support-invariant greedy extension.;
foreach $u \in \mathcal{U}$ **do**
  $m \leftarrow u$; $T \leftarrow S(u)$;
  `// extend right only if the support set is preserved`
  while $\exists c \in \Sigma$ *such that* $S(m \cdot c) = T$ **do**
    choose c; $m \leftarrow m \cdot c$;
  `// extend left only if the support set is preserved`
  while $\exists c \in \Sigma$ *such that* $S(c \cdot m) = T$ **do**
    choose c; $m \leftarrow c \cdot m$;
  add candidate pair $(m, S(m))$; `// here` $S(m) = T$ `by construction`

3) Deduplicate exact repeats.;
Collapse identical candidate pairs $(m, S(m))$.;

4) Prune within identical support sets.;
foreach *distinct support set* T *among candidates* **do**
  $\mathcal{M}_T \leftarrow \{\, m :\ (m, T)$ is a candidate $\}$;
  discard any $m \in \mathcal{M}_T$ for which $\exists m' \in \mathcal{M}_T$ such that m is a strict contiguous substring of m';

Return $\mathcal{M} = \bigcup_{T:\,|T|\geq\tau} \{(m, T) : m \in \mathcal{M}_T\}$.;

Example (why nested motifs can coexist).;
Suppose $m_A =$ ABCDEF and $m_B =$ ABCDEFGH with $m_A \subsetneq m_B$.;
If $S(m_A) = \{A, B, C\}$ but $S(m_B) = \{A, B\}$, then m_A is not removed by m_B because substring pruning is applied only within a fixed support set.;
Thus both $(m_A, S(m_A))$ and $(m_B, S(m_B))$ can remain in $\mathcal{M}$.;

B Model training

Hyperparameter	Values searched
Learning rate	5×10^{-4}, 3×10^{-4}, 2×10^{-4}, 1×10^{-4}
Batch size	16, 32, 64
Context length	256, 384, 512
Embedding dimension	256, 384, 512
Transformer layers	6, 8
Attention heads	8
Epochs	50
Seed	7

Table 1: Hyperparameter grid used for subtype-specific grid search. Best configuration per subtype selected by minimum validation loss.

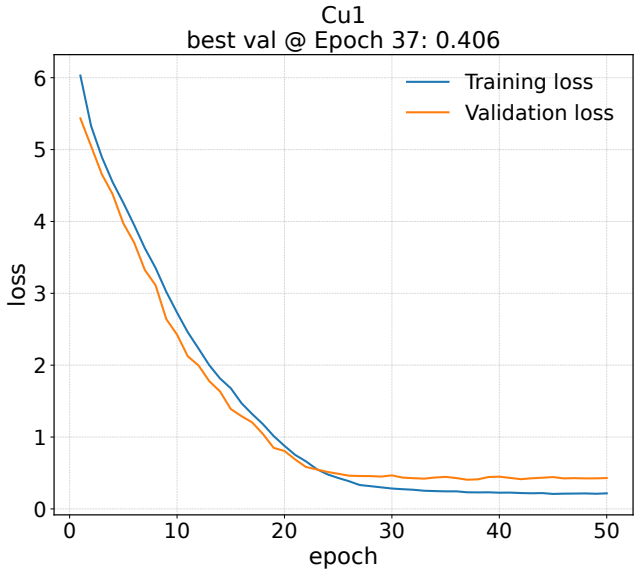

Figure 9: Training and validation loss curves for the Cu(I)-specific transformer using the optimal hyperparameter set.

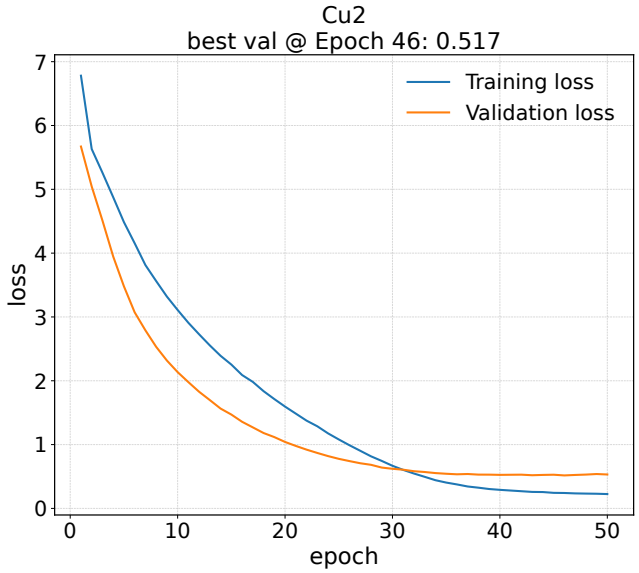

Figure 10: Training and validation loss curves for the Cu(II)-specific transformer using the optimal hyperparameter set.

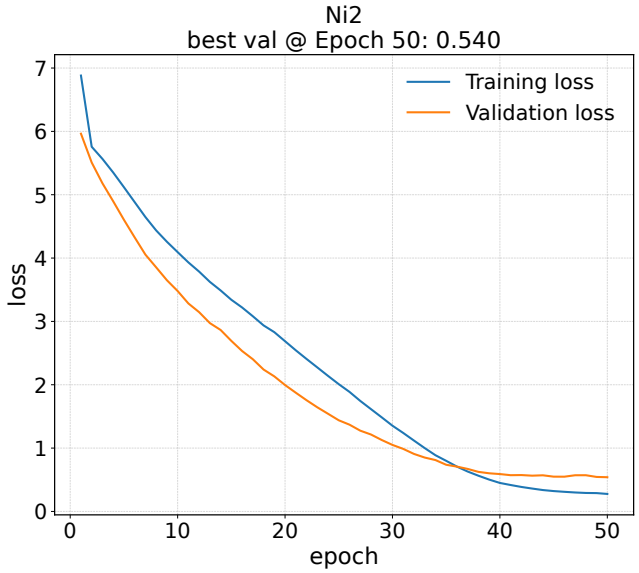

Figure 11: Training and validation loss curves for the Ni(II)-specific transformer using the optimal hyperparameter set.

C Motif-level vs. Amino-acid-Level Generation Stratified by Subtypes

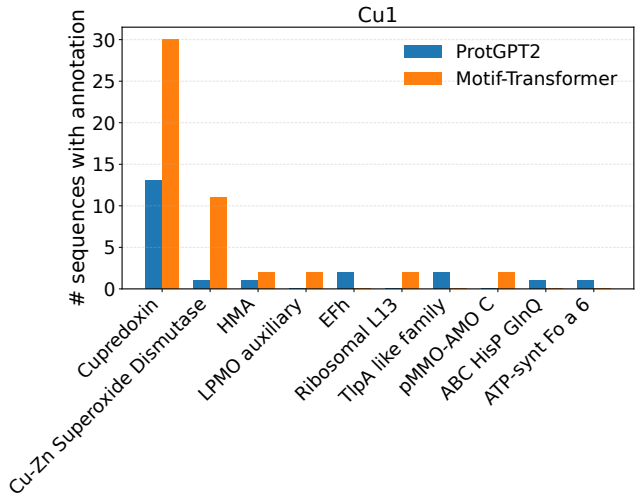

Figure 12: CDD annotations of protein sequences generated by motif-guided masking vs. our motif-level modeling for Cu(I) sequences

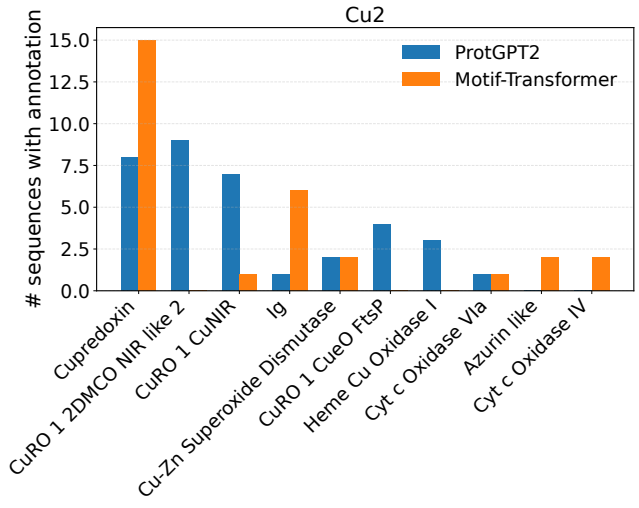

Figure 13: CDD annotations of protein sequences generated by motif-guided masking vs. our motif-level modeling for Cu(II) sequences

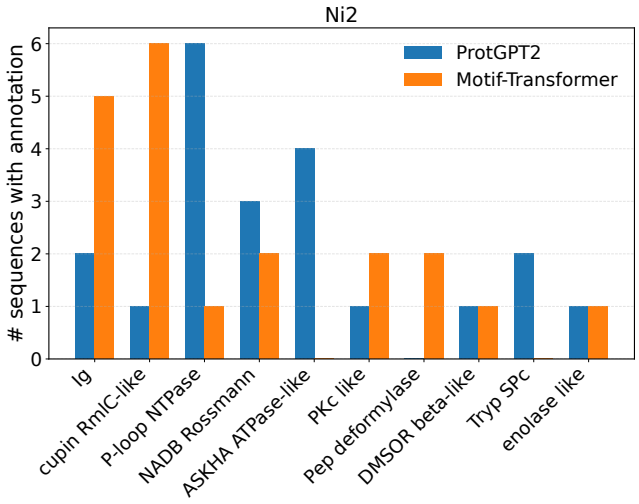

Figure 14: CDD annotations of protein sequences generated by motif-guided masking vs. our motif-level modeling for Ni(II) sequences

D Generated sequences used for structural validation

Bolded regions are motifs generated by motif-transformer.

Left Structure

MNLVTLL**QGVGQSIG**SSSLP **PKASM**LLL**SPPHLKA**LLAAL EKNSKEGRLDVIEIAHGLLW **DFKRN**LVPPGLEFEPHGSPL **FPTWHRVYV**AAYER**VIVQDA REVDS**NSETKD**VTLPYWD**YT SP**AIYWGAS**GP**GFRCAAD**LL **KDKALM**GKLEV**LVDHSI**GPG GRRPTVRDGPLK**DVIHSW**SD **AFGVK**GGGGG**HAFWF**GNGIE PGGA**VILAVWDV**GGFGN**VIV QD**LLGLQ**VLYTM**LL**FWFMEE LF**

Center Structure

CFEMLAMNDASAAAAA**GTNS AF**AKPAFAKPAFAKKVVADA AKAADENNEEAAGAERRERA AANAILAPKAAAAAGPEGQA AAAA**AGLGAGGHWDPK**AGGG L**VMRNADGSWISVVS**GEPGK PTAGTPGVCQIPGCNGTVPA SARREGRSQRAAAAIFSTLD A**ITGNAGGR**ASALLEEVNRA L**RPDKTL**AAAAG**GNWPI**AAA KLIGKKGATAED**AQLWF**GA**V CAAC**AKKLGFTPEKEGPRGY NSASASARREGRSQ

Right Structure

PGAKETIVRR**FEVSP**GG**STP PH**RH**WEHEI**YI**LEGSM**TLTS **LEGSMGERSMGL**VPGDFV**VL PDQ**LPHRVSNTSED**GEAIF**A VIPRGEGGGLGRFLVVGGGV EPEYTSAERFLVVLPLLLVG AVLIG**LYFYL**RRRGRRASFL LRVAIILV**LSIMAG**LLRIDG ALVQPSAPLL**LYFYL**LGLS**I GLLLL**ALEDSDLVGNMILGR R**GAPAEA**PEKL