PNAS RESEARCH ARTICLE | APPLIED MATHEMATICS OPEN ACCESS

# Sparse autoencoders uncover biologically interpretable features in protein language model representations

Onkar Gujral[a,b], Mihir Bafna[b], Eric Alm[c,d], and Bonnie Berger[a,b,1]

Affiliations are included on p. 12.

Contributed by Bonnie Berger; received April 20, 2025; accepted July 7, 2025; reviewed by Nir Ben-Tal and Teresa M. Przytycka

**Foundation models in biology—particularly protein language models (PLMs)—have enabled ground-breaking predictions in protein structure, function, and beyond. However, the "black-box" nature of these representations limits transparency and explainability, posing challenges for human–AI collaboration and leaving open questions about their human-interpretable features. Here, we leverage sparse autoencoders (SAEs) and a variant, transcoders, from natural language processing to extract, in a completely unsupervised fashion, interpretable sparse features present in both protein-level and amino acid (AA)-level representations from ESM2, a popular PLM. Unlike other approaches such as training probes for features, the extraction of features by the SAE is performed without any supervision. We find that many sparse features extracted from SAEs trained on protein-level representations are tightly associated with Gene Ontology (GO) terms across all levels of the GO hierarchy. We also use Anthropic's Claude to automate the interpretation of sparse features for both protein-level and AA-level representations and find that many of these features correspond to specific protein families and functions such as the NAD Kinase, IUNH, and the PTH family, as well as proteins involved in methyltransferase activity and in olfactory and gustatory sensory perception. We show that sparse features are more interpretable than ESM2 neurons across all our trained SAEs and transcoders. These findings demonstrate that SAEs offer a promising unsupervised approach for disentangling biologically relevant information present in PLM representations, thus aiding interpretability. This work opens the door to safety, trust, and explainability of PLMs and their applications, and paves the way to extracting meaningful biological insights across increasingly powerful models in the life sciences.**

interpretability | protein language models | sparse autoencoders | transcoders

Protein language models (PLMs) (1–3) are trained in an unsupervised manner on large datasets of protein sequences. A popular approach is through masked language modeling, where specific tokens are masked and the model aims to predict the masked token (1, 2). After pretraining in this unsupervised fashion, protein-level representations from these models are frequently extracted for use on downstream protein-related tasks (4, 5). Transformer-based PLMs like ESM2 (2) commonly yield $n \times d$ shaped representations for any layer, where $n$ is the length of the protein sequence and $d$ is the size of the model's vector for each position of the sequence. The protein-level representations are typically derived through mean-pooling (over the sequence dimension $n$) of the amino acid-level (token-level) representations (4), effectively condensing sequence-level information into a (fixed-length) $d$-dimensional vector for each protein. Although this approach facilitates efficient downstream applications, these representations are hard to interpret.

This lack of interpretability creates a "black-box" problem: Protein-level representations may yield high accuracy on downstream tasks, but provide little insight into what biological features are encoded in each representation and why it leads to specific downstream predictions. Understanding these representations is crucial to explainability and fostering human–AI collaboration in such downstream tasks. On the other hand, the ability to interpret amino acid-level (token-level) representations would provide a window into the internal operations of PLMs and uncover biological attributes that these models track. As PLMs continue to grow more powerful, the ability to make sense of both protein-level and amino acid-level representations will become increasingly valuable for both understanding downstream predictions and extracting meaningful biological insights, as well as ensuring that open-sourcing these models does not inadvertently expose potentially harmful biological information.

## Significance

Interpreting representations derived from protein language models (PLMs) is crucial for improving trust in the model, explainability, and human–AI collaboration in downstream applications. We leverage sparse autoencoders and transcoders from natural language processing as a way to extract biologically meaningful, interpretable features from both protein-level and amino acid-level representations. Our Gene Ontology Analysis and automated interpretability protocols uncover many sparse features that are strongly associated with specific functional annotations and protein families. We show that the sparse features are more interpretable than PLM neurons. These insights not only enhance our understanding of biological information PLMs encode and provide a pathway for gaining functional meaning from PLMs, but also enable interpretability for downstream tasks that rely on these representations.

Author contributions: O.G., M.B., E.A., and B.B. designed research; performed research; contributed new reagents/analytic tools; analyzed data; and wrote the paper.

Reviewers: N.B.-T., Tel Aviv University; and T.M.P., NIH.

The authors declare no competing interest.



[1]To whom correspondence may be addressed. Email: bab@mit.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2506316122/-/DCSupplemental.

Published August 19, 2025.

Several studies have sought to understand how PLMs encode biological information. One line of exploration has shown that models like ESM2 predict protein structures by leveraging coevolutionary statistics of interacting motifs, instead of explicitly having captured the physics of protein folding (6). Another approach has analyzed transformer-based protein models by assessing how attention heads and internal embeddings capture specific features, such as contact maps and binding sites, showing that deeper layers of the model capture more complex properties (7). Although these studies provide insights into how PLMs generally store information and the kind of information they contain, they do not offer an unsupervised method to extract the biologically meaningful features hidden in a particular PLM representation. We aimed to do exactly that, by leveraging an unsupervised approach that systematically disentangles individual PLM representations (at both the amino acid and protein-level) into interpretable features. An unsupervised approach is desirable with this kind of interpretability, as one may not always know in advance what to look for.

We draw inspiration from a very recent breakthrough in the natural language setting with the introduction of sparse autoencoders. A challenge in neural network interpretability is that neurons in general (8) [and in language models in particular (9–11)] are polysemantic: Each neuron is activated by multiple, unrelated features. The reason behind this is that the features they represent in the real world occur very sparsely, so there are a lot more distinct features that the model needs to express than there are neurons in a layer. So instead of dedicating a full neuron to each sparsely occurring feature, a layer of a neural network tries to accommodate a lot more sparsely occurring features than it has neurons in the layer by storing these features in superposition. Although this leads to a well-performing neural network, it has proven to be a significant challenge for interpretability. Passing a natural language model's internal representation through a sparse autoencoder was shown to help extract interpretable sparse features from the model's representation in an unsupervised manner (9–13).

In this work, we apply sparse autoencoders (SAEs) to PLMs to interpret PLM representations. We train SAEs on both protein-level and amino acid-level representations extracted from multiple layers of ESM2 and analyze the sparse features (i.e. neurons of the SAE's hidden layer) they uncover. To interpret these features, we conduct Gene Ontology (GO) enrichment analysis on proteins that strongly activate specific sparse features in SAEs trained on protein-level representations and identify meaningful associations with diverse biological functions, from fundamental metabolic pathways like iron-sulfur cluster assembly to specialized functions such as viral translational frameshifting.

Once the sparse features have been extracted in an unsupervised manner by our SAE, we also leverage a scalable, automated large language model (LLM)-assisted interpretability approach, as opposed to humans, to interpret the sparse features. We find that many sparse features correspond to biologically meaningful entities like protein families and functions, which include the NAD Kinase family involved in NAD metabolic processes and NADP biosynthesis, the IUNH family linked to nucleoside hydrolase activity, and the PTH family. Notably, sparse features are more interpretable than standard ESM2 neurons across all tested layers.

Further, we introduce transcoders (14) to PLMs to learn a sparsified approximation of the transformation from the protein-level representation in one layer of ESM2 to the next, with interpretable sparse features. These transcoder-derived sparse features exhibit interpretability on par with those from SAEs, suggesting that they offer an alternative way to understand the organization of information within individual PLM representations. Collectively, our findings demonstrate that SAEs and their variants disentangle complex PLM representations into biologically interpretable features in an unsupervised fashion.

By disentangling biologically relevant features in an unsupervised manner from protein-level representations, our approach has broad implications for explainability in the context of downstream protein-level tasks. At the same time, this approach also opens up the potential to gain fresh biological insights from systematically understanding interpretable features that PLMs track in AA-level representations. Moreover, as foundational models get more advanced, we expect SAEs and their variants to play an increasingly significant role in their interpretability.

## Results

### Overview of Method.

***Sparse autoencoders.*** A sparse autoencoder (9–13, 15, 16) is an autoencoder, with a single hidden layer, but instead of the hidden layer being much narrower than the input, it is much wider; this autoencoder is constrained to activate the neurons of the hidden layer in a sparse fashion on any given input. A sparse autoencoder is trained on the activations of a chosen layer of a model. The result is that the wide hidden layer has sparsely activating neurons, thereby disentangling polysemantic neurons into sparse neurons that, in the NLP setting, have been shown to be more monosemantic (therefore more interpretable) than the original neurons. These sparsely activating neurons are referred to as sparse features or SAE features throughout the paper. Several groups have used sparse autoencoders to interpret large language models trained on natural language (9–13). They found several sparse features that capture interpretable features in natural language. For instance, they find a feature that activates on Hebrew script, another feature that captures DNA sequences, and yet another one that captures mathematical symbols (9).

***Our workflow.*** Inspired by the success of sparse autoencoders in the NLP setting, here we leverage sparse autoencoders to interpret PLMs. We train SAEs to interpret i) protein-level representations and ii) amino acid-level representations from various layers of the ESM2 model: "esm2 t12 35M UR50D." Fig. 1*A* illustrates the two points in the process where we may attempt to interpret PLM representations. The first is interpreting the protein-level representations from ESM2, which is helpful from a downstream perspective since these representations get used on downstream protein-level tasks. The second is interpreting the amino acid-level representations from ESM2, which is helpful for understanding what features ESM2 seems to be tracking.

Although there are multiple ways to extract protein-level representations from ESM2, a very commonly used and often superior way tends to be mean-pooling (4), so we employ it in this work. We trained SAEs (schematic in Fig. 1*B*) to extract features (in an unsupervised fashion) from the protein-level representations and amino acid-level representations from layers 6, 7, 8, 9, 10 of ESM2, and then aimed to interpret these sparse features in the context of the following metadata: protein names, protein families, gene names, and GO terms, including biological processes, cellular components, and molecular functions. See *Materials and Methods* for more details on the sparse autoencoder architecture and interpretation methodology.

***Transcoders.*** We also train a variant of sparse autoencoders called transcoders [originally proposed by Dunefsky et al. in the NLP

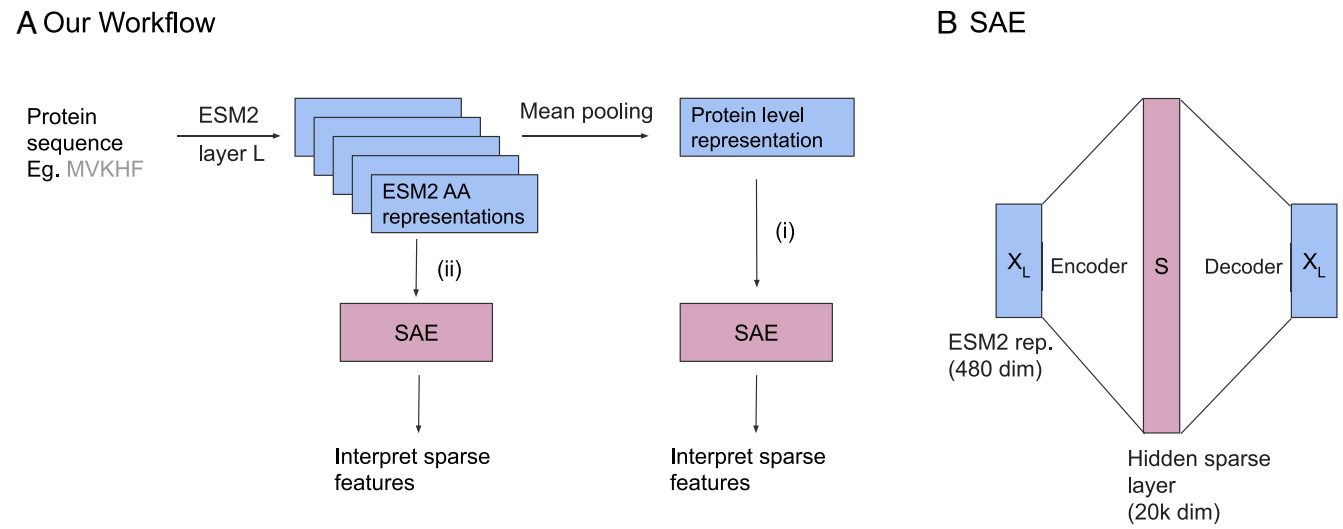


**Fig. 1.** Interpreting ESM representations with SAEs. (*A*) Brief illustration of our workflow highlighting the two places at which we choose to interpret representations from ESM2 with the help of a sparse autoencoder (SAE): i) at the protein level by passing the single protein-level representation through the SAE—since these representations get used on a lot of downstream tasks, interpreting them is helpful for human–AI collaboration and explaining performance on downstream tasks. ii) at the amino acid-level, by passing individual amino acid (AA) level representations through the SAE one at a time, allowing us to understand what features ESM2 is tracking. The blue boxes are representations drawn from the ESM2 model (at the protein-level or AA-level respectively); these are represented as $X_L$ in (*B*). (*B*) Schematic of the sparse autoencoder (SAE) with a hidden dimension of 20,000 trained on $X_L$, which is either the 480-dimensional protein-level representation, or 480-dimensional individual AA-level representation from layer L of ESM2.

setting (14)] on the protein-level representations. Here, the input and sparsity constraint on the hidden layer is the same as for our sparse autoencoder, but the output that it aims to predict is the protein-level representation of the next layer of the PLM. So these transcoders are made to learn a sparse approximation of the computation to go from one layer's protein-level representation to the next. The resulting sparse features are interpreted similarly to those from our SAEs.

We trained sparse autoencoders and transcoders on protein-level and amino acid-level representations extracted from layers 6, 7, 8, 9, 10 of ESM2.*

**GO Analysis Uncovers Strong Associations of Sparse Features with GO Terms.** To investigate the biological interpretability of the sparse features learned from the SAEs trained on protein-level representations, we aimed to analyze their association with functional annotations. We work with a dataset of 18,142 randomly selected human proteins from the UniProt dataset (17) for this analysis.† For each sparse feature, we identified the set of proteins in our dataset for which the feature was activated. All proteins with a nonzero activation value for a specific feature were included in the candidate set of proteins for that feature.

We performed a GO (18) enrichment analysis for each set of highly activated proteins. GO annotations provide a standardized vocabulary to describe the molecular functions, biological processes, and cellular components associated with proteins. Using the set of proteins corresponding to a sparse feature, we tested for overrepresentation of GO terms relative to the background set of all proteins in the dataset. Enrichment analysis was conducted using a hypergeometric test, and *P*-values were corrected for multiple hypothesis testing using the Benjamini–Hochberg procedure to control the false discovery rate.

The results of the GO enrichment analysis revealed that certain sparse features were strongly associated with niche functional annotations. For instance, one feature was highly activated in proteins enriched for "retrotransposition" (GO:0032197) and "transposition" (GO:0032196), while another feature was associated with "iron-sulfur cluster assembly" (GO:0016226) and "metallo-sulfur cluster assembly" (GO:0031163), functions characteristic of specific metabolic processes. Importantly, many of these enriched terms represented distinct biological functions or processes, such as "pentose-phosphate shunt, oxidative branch" (GO:0009051) and "positive regulation of protein oxidation" (GO:1904808), suggesting that the sparse autoencoder successfully disentangled biologically meaningful patterns from the original protein-level ESM representations.

To further assess the interpretability of the sparse features, we aimed to evaluate two key properties of the protein sets associated with each feature. First, we sought to determine whether certain features activated only for very specific functions, as indicated by GO terms that were proximal in the GO Directed Acyclic Graph (DAG) (18). Such specificity would not only imply that the feature was capturing meaningful functional attributions for the proteins that activated it, but also suggest that the sparse feature was learning a more "monosemantic" representation; by this we mean a feature that favors GO terms that are closer in the GO DAG and use that as a proxy for GO monosemanticity here. To quantify this, for each sparse feature, we took the set of GO terms it enriched (with significance $\leq 0.05$) and computed the pairwise shortest path lengths between all terms in the original GO DAG. The mean shortest path length was then calculated per sparse feature. A smaller mean shortest path length indicated that the enriched GO terms were more proximal in the DAG, reflecting a higher degree of GO monosemanticity for the feature.

After understanding the GO monosemanticity of each sparse feature, we then wanted to assess where on the GO DAG these monosemantic features mapped. To do this, for each sparse

*The motivation for interpreting these layers of ESM2 is that the ESM2 model we work with has a total of 12 layers, and drawing on intuition from interpretability work on LLMs (10), we would expect the most interesting layers for interpretation to be those somewhere near the end of the model, however not absolutely at the end—since those are likely to be have been specialized for the unsupervised model training task.

†In total there are 26467, 10146, 4022 GO terms in the GO directed acyclic graph (DAG) belonging to the 3 namespaces: biological process, molecular function, and cellular component, resp. Of these, 11545, 4354, 1713 GO terms appear in our set of GO terms used in the protein set used for our GO analysis. Thus, for the 3 namespaces, 43.62%, 42.91%, 42.59% of the GO terms, resp., are covered by our protein set used for the GO analysis.

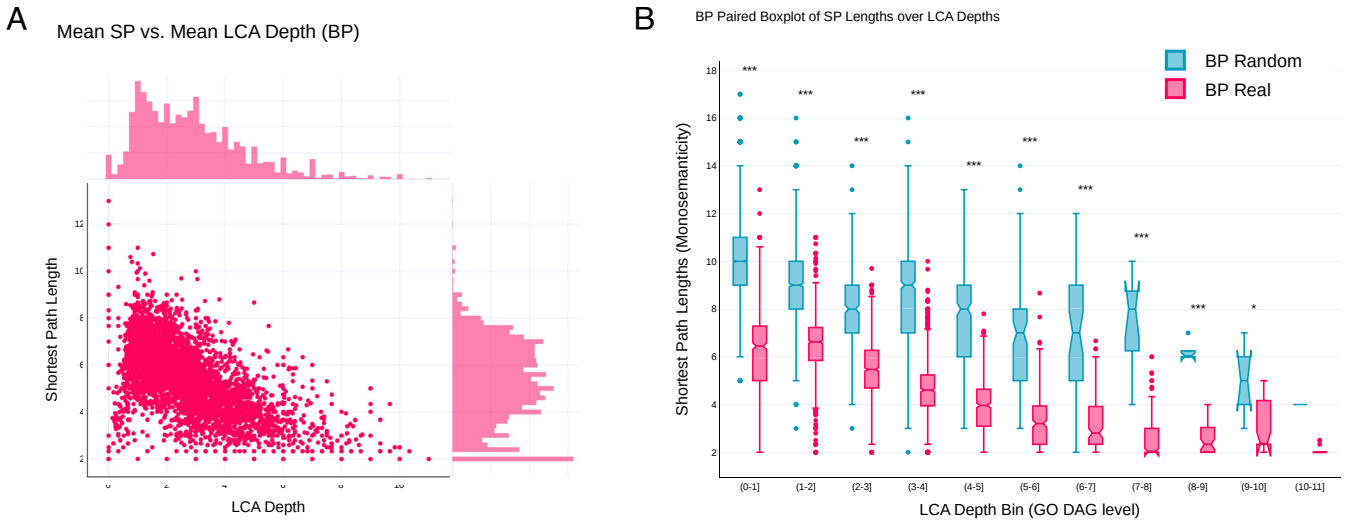


**Fig. 2.** Evidence of biological process specific neurons. Each neuron (point) is associated with a collection of proteins that it activates. GO Enrichment was conducted for each of these protein sets to associate GO terms per neuron. (*A*) Scatterplot shows a comparison of the average pairwise shortest path length and least common ancestor among each neuron's enriched GO Terms. Significant number of neurons have low shortest path lengths (monosemantic) across all levels of the GO DAG. (*B*) Grouped boxplot shows comparison to random pairwise shortest path lengths over each least common ancestor (LCA) depth. All results shown for sparse features from SAE trained on protein-level representations of ESM2 Layer 7.

feature's enriched set of GO terms, we computed each pairwise least common ancestor (LCA). The mean of the depths of these LCAs was calculated to indicate the approximate position or depth (distance from root) of the GO terms enriched by that sparse feature. This metric provided a quantitative understanding of whether the sparse feature was associated with more general (higher-level) or specific (lower-level) functional annotations.

These two metrics-GO monosemanticity and position on the GO DAG-allowed us to characterize individual sparse features. Notably, we observed strong evidence of neurons that are functionally monosemantic (by our definition above) across all depths of the GO hierarchy as indicated by the low shortest path lengths across all LCA depths in Fig. 2*A*. Examples of these neurons and associated GO Terms are organized in Table 1. To understand the significance of this result, we established a random baseline by computing pairwise shortest path lengths and LCA depths for randomly sampled GO terms that were linked with the specific set of human proteins that we used. At every depth of the GO DAG (LCA Depth), the pairwise shortest path lengths of each neuron's GO terms were significantly smaller than random (Fig. 2*B*). Analogous figures for cellular component and molecular function respectively are in *SI Appendix*, Figs. S1–S4.

With this analysis, we demonstrated that certain sparse features could be mapped to the GO DAG across all levels of the hierarchy, from very specific leaf-level nodes to broader functional categories. Notably, this mapping was achieved in an entirely unsupervised manner, leveraging only the information extracted from the ESM protein-level representations, without any direct supervision or prior GO knowledge.

Finally we remark on the extent to which GO terms in the DAG and the protein set used for GO analysis are covered by sparse features (neurons). The total number of enriched GO terms for layer 9's SAE is 9789. These are partitioned into 6024, 2740, and 1025 biological process, molecular function, and cellular component GO terms, resp. In total there are 26467, 10146, and 4022 GO terms in the DAG belonging to the three namespaces biological process, molecular function, and cellular component, resp. Of these, 11545, 4354, 1713 GO terms appear in our set of GO terms used in the protein set used for our GO analysis; thus, for layer 9's SAE, 52.17%, 62.93%, 59.83% of the GO terms present in the set used for GO analysis feature among the set of GO terms enriched by neurons. *SI Appendix*, Table S3 provides this information for all other layers and models. We also investigate the degree of overlap between enriched GO terms found by SAEs and transcoders for the same layer. For example, in layer 9's SAE, we find 9789 enriched GO terms whereas for layer 9's transcoder we find 9659 enriched GO terms. Of these, the two share 9180 enriched GO terms in common. Thus, both methods (SAEs and transcoders) have a majority of their enriched GO terms in common. *SI Appendix*, Table S4 indicates the degree of overlap for all layers.

**LLM-Assisted Interpretation Finds Biologically Relevant Sparse Features.** Once we have our sparse features from the SAEs and transcoders, we draw from work in natural language processing (9, 11, 19, 20) to automate the interpretation of these sparse features using an LLM; automation is critical due to the large number of sparse features. We do this by selecting a sparse feature (neuron) from among 20,000 that is sufficiently active on a 50,000 sequence subset of UniProt. We pick 17 active and 15 inactive sequences for this sparse feature and as context we provide Claude[‡] with half of these sequences' ID, activation value, protein names, gene names, protein families, and GO terms, including biological processes, cellular components, molecular functions—and asked Claude to come up with a single sentence long human-readable interpretation for what our SAE's sparse feature[§] appears to be capturing. We aimed to assess the interpretability of sparse features in the context of this metadata. So we then score the Claude-provided feature interpretation by first asking Claude (in a separate chat) to use this interpretation to predict (i.e. simulate) the feature's activations using the same metadata for the withheld half of the sequences from above. Then following work in the NLP setting (9, 11, 19), we compute the Pearson correlation between the actual activation values and the simulated ones that Claude predicts, and this is regarded as the interpretability

[‡]Claude 3.5 Sonnet.

[§]The SAE's sparse features are simply neurons in the SAE's hidden sparse layer.

**Table 1. Examples of sparse neuron's associated with specific biological processes across all levels of the GO DAG**

| Neuron | LCA depth | Terms |
|---|---|---|
| 2793 | 1 | ["viral translational frameshifting (GO:0075523)," "viral process (GO:0016032)," "aspartic-type endopeptidase activity (GO:0004190)," "aspartic-type peptidase activity (GO:0070001)"] |
| 10282 | 2 | ["retrotransposition (GO:0032197)," "transposition (GO:0032196)"] |
| 9095 | 3 | ["peptide modification (GO:0031179)," "peptide metabolic process (GO:0006518)"] |
| 16984 | 4 | ["positive regulation of viral budding via host ESCRT complex (GO:1903774)," "regulation of viral budding via host ESCRT complex (GO:1903772)"] |
| 7631 | 5 | ["metallo-sulfur cluster assembly (GO:0031163)," "iron-sulfur cluster assembly (GO:0016226)"] |
| 953 | 6 | ["protein maturation by [4Fe-4S] cluster transfer (GO:0106035)," "protein maturation by iron-sulfur cluster transfer (GO:0097428)"] |
| 1105 | 7 | ["iron ion transport (GO:0006826)," "transition metal ion transport (GO:0000041)"] |
| 8646 | 8 | ["positive regulation of protein oxidation (GO:1904808)," "regulation of protein oxidation (GO:1904806)"] |
| 3823 | 9 | ["leukotriene D4 catabolic process (GO:1901749)," "leukotriene catabolic process (GO:0036100)," "dipeptidase activity (GO:0016805)," "metallodipeptidase activity (GO:0070573)"] |
| 11233 | 10 | ["positive regulation of serine-type endopeptidase activity (GO:1900005)," "positive regulation of serine-type peptidase activity (GO:1902573)"] |
| 3771 | 11 | ["pentose-phosphate shunt, oxidative branch (GO:0009051)," "NADPH regeneration (GO:0006740)," "glucose-6-phosphate dehydrogenase activity (GO:0004345)"] |

Only sparse neurons whose average shortest path length among its enriched GO Terms was $\leq 2$ were chosen. One sparse neuron per least common ancestor (LCA) Depth (GO DAG Level) was selected, and the higher the LCA Depth, the more niche the biological process. Table was created using sparse neurons (features) from the SAE trained on the protein-level representation of ESM2 Layer 7.

score for the feature. Using Claude for this enables a scalable method of scoring a feature's interpretability. We employ this autointerpretability method for sparse features from all our SAEs and transcoders (See *Materials and Methods* for more details.)

***SAE results for protein-level representations.*** We demonstrate that many sparse features extracted from the SAEs trained on protein-level representations are very interpretable, as evidenced by the Pearson correlation scores in Table 2. Across all examined layers, the median Pearson correlations are near or above 0.7. Table 3 presents examples of Claude's automated interpretations for both highly interpretable (0.9+ scoring) and near-median level interpretable sparse features. For instance, a sparse feature (2579 in layer 7) is activated by proteins with methyltransferase activity, especially in RNA and DNA methylation. Additionally, certain sparse features (11982 in layer 9, 13811 in layer 7, and 19647 in layer 8) are linked to specific protein families and their roles, such as the NAD kinase family (NAD metabolic processes and NADP biosynthesis), Diaminopimelate epimerase family (lysine biosynthesis via diaminopimelate with diaminopimelate epimerase activity) and the IUNH family (nucleoside hydrolase activity). Another sparse feature (16397 in layer 10) is associated with proteins involved in pseudouridine synthesis, especially tRNA and rRNA pseudouridine synthesis, from the TruB and RluA families, while sparse feature 15683 in layer 10 is linked

**Table 2. Comparison of sparse autoencoder (SAE), Transcoder (TC), and ESM neurons autointerpretability scores for various layers for the protein-level representations**

| Model (Layer) | Median correlation | Mean correlation | % with $P < 0.05$ | Neurons analyzed | Neurons attempted |
|---|---|---|---|---|---|
| Sparse (layer 6) | 0.706 | 0.634 | 76% | 200 | 819/20,000 |
| TC (layer 6) | 0.710 | 0.635 | 78.5% | 200 | 882/20,000 |
| ESM (layer 6) | 0.52 | 0.443 | 52.8% | 200 | 202/480 |
| Sparse (layer 7) | 0.702 | 0.629 | 72% | 200 | 703/20,000 |
| TC (layer 7) | 0.734 | 0.640 | 75.5% | 200 | 598/20,000 |
| ESM (layer 7) | 0.471 | 0.418 | 48.2% | 200 | 201/480 |
| Sparse (layer 8) | 0.726 | 0.623 | 73% | 200 | 585/20,000 |
| TC (layer 8) | 0.704 | 0.649 | 77.5% | 200 | 575/20,000 |
| ESM (layer 8) | 0.454 | 0.42 | 49.5% | 200 | 202/480 |
| Sparse (layer 9) | 0.692 | 0.618 | 69% | 200 | 684/20,000 |
| TC (layer 9) | 0.682 | 0.638 | 76% | 200 | 565/20,000 |
| ESM (layer 9) | 0.425 | 0.398 | 43% | 200 | 203/480 |
| Sparse (layer 10) | 0.756 | 0.659 | 76% | 200 | 601/20,000 |
| TC (layer 10) | 0.667 | 0.601 | 69% | 200 | 629/20,000 |
| ESM (layer 10) | 0.435 | 0.383 | 42% | 200 | 201/480 |

Sparse and TC neurons are more interpretable than ESM neurons. (Note that for Sparse and TC there are 20,000 neurons to pick from to analyze and for ESM there are 480 neurons to pick from, but we only analyze a neuron's interpretability if it is sufficiently active. This is why there are more neurons attempted than analyzed especially for Sparse and TC.) Here, *P*-value means *P*-value of the Pearson correlation (i.e. the autointerpretability score), so that column indicates the percentage of neurons with $P < 0.05$.

**Table 3. Examples of several SAE neurons (from various layers) with very high interpretability and some median interpretability, with Claude's interpretation, for protein-level representations**

| Model | Neuron no. | Claude's interpretation | Correlation |
|---|---|---|---|
| Sparse (layer 9) | 6384 | This neuron appears to be primarily detecting proteins involved in phosphoenolpyruvate carboxykinase (ATP) activity and gluconeogenesis, with a strong focus on the Phosphoenolpyruvate carboxykinase (ATP) family. | 0.972 |
| Sparse (layer 9) | 11982 | This neuron appears to be detecting proteins that belong to the NAD kinase family and are involved in NAD metabolic processes and NADP biosynthesis. | 0.954 |
| Sparse (layer 7) | 13811 | This neuron appears to be detecting proteins belonging to the Diaminopimelate epimerase family, specifically those involved in lysine biosynthesis via diaminopimelate with diaminopimelate epimerase activity. | 0.993 |
| Sparse (layer 7) | 2579 | This neuron appears to be detecting proteins with methyltransferase activity, particularly those involved in RNA or DNA methylation processes. | 0.930 |
| Sparse (layer 8) | 19647 | This neuron appears to be detecting proteins with nucleoside hydrolase activity, particularly those belonging to the IUNH family and involved in nucleobase metabolic processes. | 0.999 |
| Sparse (layer 8) | 3800 | This neuron appears to be detecting proteins involved in transmembrane transport of ions or amino acids, particularly those located in the plasma membrane. | 0.721 |
| Sparse (layer 10) | 16397 | This neuron appears to be detecting proteins involved in pseudouridine synthesis, particularly tRNA and rRNA pseudouridine synthases from the TruB and RluA families. | 0.969 |
| Sparse (layer 10) | 15683 | This neuron appears to be detecting proteins involved in the electron transport chain, particularly NADH dehydrogenase and quinone oxidoreductase components of Complex I. | 0.758 |

The correlation score, ranging between 0 and 1, represents the interpretability of the SAE neuron. See *SI Appendix*, section S5 for full list.

to Complex I proteins of the electron transport chain. These findings describe the ability of SAEs to disentangle meaningful biological signals from protein-level representations in an entirely unsupervised manner. This enhances the interpretability of these representations that frequently get used in downstream protein-level tasks.

We compare the automated interpretability scores of sparse features (Table 2) with those of ESM neurons in the protein-level representation (see *Materials and Methods* for details). Across the examined layers, ESM neurons exhibit median Pearson correlations slightly above 0.5 at best, whereas sparse features consistently achieve median correlations around 0.7. This demonstrates that the SAE-derived features are more interpretable than ESM neurons. Fig. 3*A* provides a visual comparison, showing an overlaid histogram of interpretability scores for sparse features extracted from the SAE trained on layer 9, alongside ESM neuron scores from the same layer. The distribution highlights a notable number of highly interpretable sparse features (with Pearson correlations between 0.7 and 1), compared to a smaller number for ESM neurons. This reinforces the advantage of SAEs in disentangling biologically meaningful information from protein-level representations.

***SAE results for amino acid-level representations.*** We also aim to functionally interpret SAE features extracted from SAEs trained on amino acid (AA) level representations. For this we follow the same autointerpretability method as we do for SAE features from the SAE trained on protein-level representations—however, since protein sequences can have a large number of amino acids and this number can be very different for different protein sequences, to sustain feasibility as in the protein-level case, for an SAE feature (or neuron in the case of ESM) we aim to interpret the mean activation of that SAE feature (or neuron) across the entire protein sequence.

We show that many sparse features from the SAEs trained on AA-level representations are also very interpretable, as demonstrated by the interpretability scores (Pearson correlations) in Table 5. The median Pearson correlations are 0.673 to 0.714 across the layers examined. Table 4 highlights several highly interpretable (around 0.9 scoring) and some near median-level interpretable sparse features extracted by SAEs trained on amino acid-level representations, demonstrating their ability to capture functionally meaningful patterns. Many of these features align closely with known protein families and functions, and at least one with an uncharacterized protein family. For example, sparse feature 19717 (layer 9) is associated with proteins from the UPF0312 family (uncharacterized), while feature 7554 (layer 10) is associated with proteins involved in olfactory and gustatory sensory perception. Certain sparse features correspond to specific enzyme families like feature 15537 (layer 9), linked to the PTH family with peptidyl-tRNA hydrolase activity and involved in translation processes, and feature 12791 (layer 7) associated with the Metallo-dependent hydrolases superfamily and Adenine deaminase family. Another sparse feature (10650 in layer 7) is linked to proteins with calcium ion binding functionality. These findings indicate that SAEs can (in an unsupervised manner) extract functionally relevant signals from amino acid-level representations, revealing several functionally distinct protein families and metabolic pathways.

We compare the automated interpretability score of sparse features in Table 5 with that of ESM neurons in the AA-level representation (see *Materials and Methods* for details). The median Pearson correlations for ESM neurons peaks just below 0.5 across examined layers, whereas SAE features achieve a median score of 0.673 to 0.714, demonstrating their superior interpretability for AA-level representations as well—and comparable to the performance for protein-level SAEs. Fig. 3*C* presents an overlaid histogram with automated interpretability scores of layer 9's SAE and layer 9's ESM neurons for the AA-level representations.

***Transcoder results for protein-level representations.*** Transcoders (14) were originally developed to create a sparse approximation

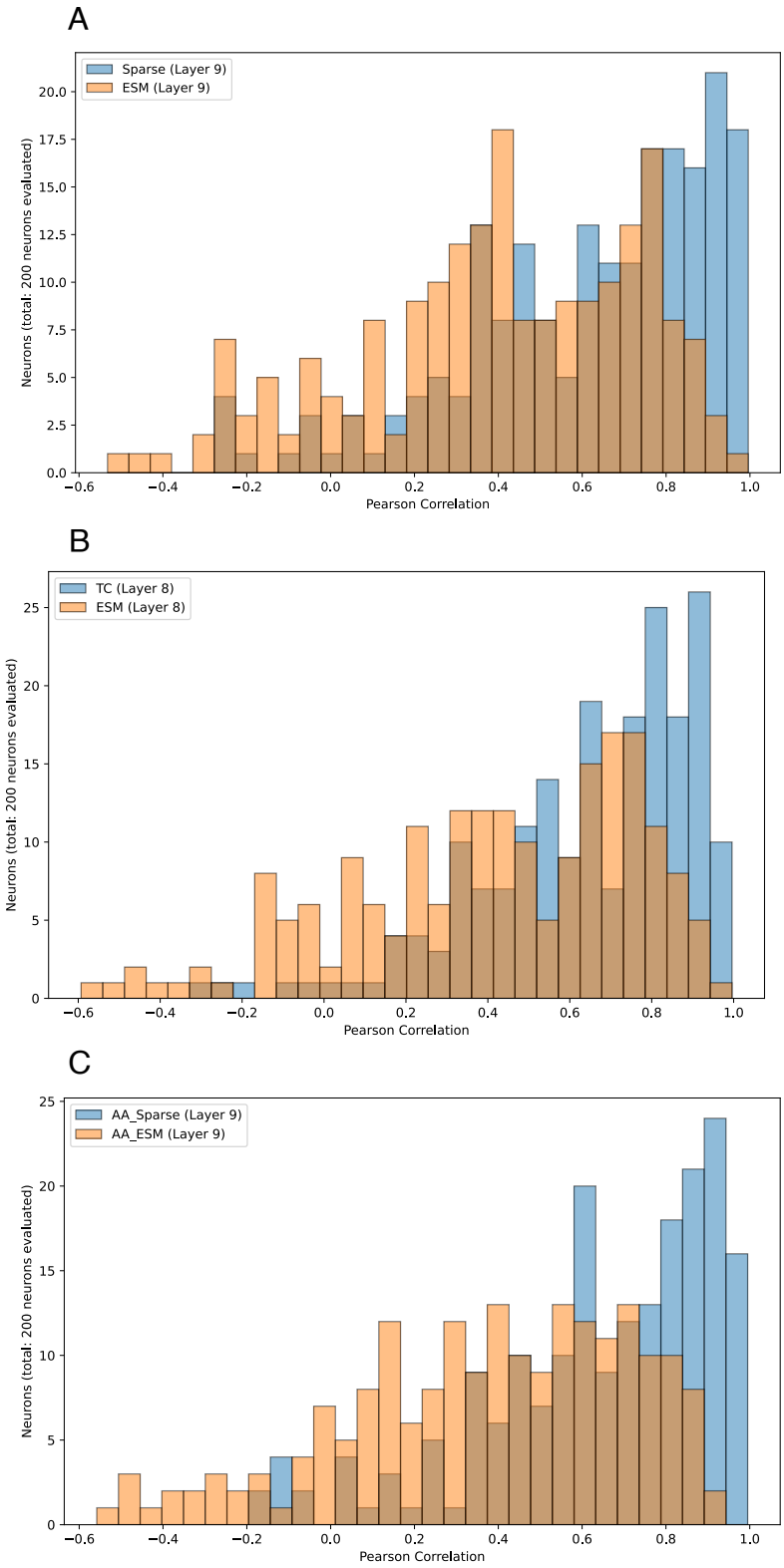


**Fig. 3.** Overlaid correlation histograms of sparse features and ESM neurons, with the overlaid portion of the histogram shaded in brown (made by using the same range for both histograms): (*A*) Sparse features from SAE (layer 9) and ESM neurons from layer 9—for protein-level representations. (*B*) Sparse features from Transcoder (TC) (layer 8) and ESM neurons from layer 8—for protein-level representations. (*C*) Sparse features from SAE (layer 9) and ESM neurons from layer 9—for AA-level representations. (2 nan values were discarded for the SAE, leaving 198 neurons).

of the multilayer perceptron sublayer of a transformer block at the token level in NLP. But we train our transcoders to learn a sparse approximation of the computation needed to get from the protein-level representation in layer L to the protein-level representation in layer L+1. So the sparse features in our transcoder are features that are involved in transforming one layer's protein-level representation into the next. The sparse features thus extracted are comparably interpretable to those from SAEs as shown in Table 2, with median interpretability scores of 0.66 to 0.73 for the layers examined. Fig. 3*B* is an overlaid histogram showing the interpretability scores for the sparse features from the transcoder trained on layer 8's protein-level representation as well as the interpretability scores of ESM neurons in layer 8's protein-level representation. It shows several highly interpretable sparse features (with Pearson correlations between 0.7 and 1) compared to ESM neurons.

**Table 4. Examples of several SAE neurons (from various layers) with very high interpretability and some median interpretability, with Claude's interpretation, for AA-level representations**

| Model | Neuron no. | Claude's interpretation | Correlation |
|---|---|---|---|
| Sparse (layer 10) | 7554 | This neuron appears to be detecting proteins involved in olfactory or gustatory sensory perception, particularly insect chemoreceptors and taste receptors. | 0.987 |
| Sparse (layer 8) | 11965 | This neuron appears to be detecting proteins with sulfotransferase activity, particularly those involved in steroid or lipid metabolism. | 0.714 |
| Sparse (layer 9) | 15537 | This neuron appears to be primarily detecting proteins with peptidyl-tRNA hydrolase activity, particularly those belonging to the PTH family and involved in translation processes. | 0.995 |
| Sparse (layer 10) | 16441 | This neuron appears to be detecting proteins that function as transmembrane transporters, particularly those belonging to the Major Facilitator Superfamily or other transporter families. | 0.711 |
| Sparse (layer 9) | 19717 | This neuron appears to be detecting proteins belonging to the UPF0312 family, specifically those located in the periplasmic space. | 0.981 |
| Sparse (layer 9) | 10718 | This neuron appears to be detecting proteins that belong to the Cation transport ATPase (P-type) (TC 3.A.3) family, particularly those involved in transporting metal ions like calcium, potassium, or copper across membranes. | 0.974 |
| Sparse (layer 7) | 10650 | This neuron appears to be primarily detecting proteins with calcium ion binding functionality, often associated with muscle-related or calcium-responsive cellular processes. | 0.967 |
| Sparse (layer 9) | 1346 | This neuron appears to be detecting proteins involved in molybdopterin cofactor biosynthesis or related enzymatic activities, particularly those with molybdopterin molybdotransferase or similar functions. | 0.990 |
| Sparse (layer 7) | 12791 | This neuron appears to be detecting proteins with adenine deaminase activity, particularly those belonging to the Metallo-dependent hydrolases superfamily and Adenine deaminase family. | 0.998 |

The correlation score, ranging between 0 and 1, represents the interpretability of the SAE neuron. See *SI Appendix*, section S5 for full list.

Table 6 presents examples of Claude's automated interpretations for both very highly interpretable (0.9+ scoring) and near-median level interpretable sparse features extracted by transcoders. Many of these features align with specific protein families and functions. For instance, sparse feature 9270 (in layer 9) is associated with proteins that function as arginine-tRNA ligases, while feature 12680 in layer 9 captures proteins involved in apoptosis, especially caspases. Additionally, feature 15097 (layer 8) is associated with proteins from the Short-chain dehydrogenases/reductases (SDR) family, particularly those involved in fatty acid elongation and retinol metabolism. Another sparse feature (4745 in layer 7) captures proteins from the chemokine family, which are involved in immune response and cell signaling processes, while sparse feature 19393 in layer 10 is associated with UDP-glycosyltransferase activity. These findings demonstrate that transcoders are able to extract sparse features that align with biologically meaningful features like protein families and functions in an unsupervised manner, underscoring their potential as an alternative method to SAEs for understanding protein-level PLM representations.

Since SAE feature interpretations tend to correspond to family and function, we expect an increased sequence similarity between coactivating proteins. We performed experiments to verify that this is the case. We first measured both the sequence similarity

**Table 5. Comparison of SAE and ESM neurons' autointerpretability scores for various layers for the amino acid-level representations**

| Model (layer) | Median correlation | Mean correlation | %age with $P < 0.05$ | Neurons analyzed | Neurons attempted |
|---|---|---|---|---|---|
| Sparse (layer 6) | 0.673 | 0.616 | 74.5% | 200 | 220/20,000 |
| ESM (layer 6) | 0.499 | 0.441 | 51.8% | 200 | 202/480 |
| Sparse (layer 7) | 0.680 | 0.593 | 73.5% | 200 | 208/20,000 |
| ESM (layer 7) | 0.442 | 0.384 | 44.5% | 200 | 201/480 |
| Sparse (layer 8) | 0.703 | 0.589 | 71% | 200 | 209/20,000 |
| ESM (layer 8) | 0.456 | 0.423 | 48% | 200 | 201/480 |
| Sparse (layer 9) | 0.714 | 0.645 | 75.8% | 200 | 206/20,000 |
| ESM (layer 9) | 0.416 | 0.380 | 44% | 200 | 201/480 |
| Sparse (layer 10) | 0.703 | 0.625 | 73.5% | 200 | 208/20,000 |
| ESM (layer 10) | 0.438 | 0.393 | 45.7% | 200 | 201/480 |

Sparse neurons are more interpretable than ESM neurons. Here, *P*-value means *P*-value of the Pearson correlation (i.e. the autointerpretability score), so that column indicates the percentage of neurons with $P < 0.05$.

**Table 6. Examples of Transcoder neurons (from various layers) with very high interpretability and near-median interpretability, with Claude's interpretation, for protein-level representations**

| Model | Neuron no. | Claude's interpretation | Correlation |
|---|---|---|---|
| TC (layer 9) | 9270 | This neuron appears to be detecting proteins that function as arginine–tRNA ligases (arginyl-tRNA synthetases) involved in arginyl-tRNA aminoacylation. | 0.989 |
| TC (layer 9) | 12680 | This neuron appears to be detecting proteins involved in apoptosis, particularly caspases and related proteins that play a role in programmed cell death pathways. | 0.680 |
| TC (layer 8) | 12186 | This neuron appears to be detecting proteins with FMN-dependent NADH:quinone oxidoreductase activity, particularly those in the Azoreductase type 1 family. | 0.937 |
| TC (layer 8) | 12187 | This neuron appears to be detecting proteins involved in metal ion binding and transport, particularly those related to iron, copper, and other metals. | 0.711 |
| TC (layer 8) | 15097 | This neuron appears to be detecting proteins belonging to the Short-chain dehydrogenases/reductases (SDR) family, particularly those involved in fatty acid elongation or retinol metabolism. | 0.747 |
| TC (layer 7) | 4745 | This neuron appears to be detecting proteins belonging to the chemokine family, particularly those involved in immune response and cell signaling processes. | 0.989 |
| TC (layer 10) | 19393 | This neuron appears to be detecting proteins with UDP-glycosyltransferase or UDP-glucuronosyltransferase activity, primarily from the UDP-glycosyltransferase family. | 0.943 |
| TC (layer 10) | 4745 | This neuron appears to be detecting proteins involved in GTP 3',8-cyclase activity and molybdenum cofactor biosynthesis, particularly those belonging to the MoaA family within the Radical SAM superfamily. | 0.673 |

The correlation score, ranging between 0 and 1, represents the interpretability of the Transcoder neuron. See *SI Appendix*, section S5 for full list.

between the set of coactivating proteins and the sequence similarity between the set of non-coactivating proteins in the set of interpretation sequences for a neuron (i.e. SAE feature) analyzed in (for example) layer 9's AA-level SAE. We find that the coactivating proteins have increased sequence similarity as measured by global alignment, local alignment, and k-mer (k = 3) scores compared to non-coactivating proteins (−0.5304, 0.4272, 0.0558) vs. (−1.7535, 0.2149, 0.0287). However, the Pearson correlation between the sequence similarity scores for a neuron and the neuron's interpretability score is weak (0.263, 0.117, 0.188)—i.e. our SAE neurons with greater sequence similarity among their coactivating proteins are not necessarily more interpretable—suggesting that there is likely more to an interpretable SAE neuron than increased sequence similarity. Next, for layer 9's protein-level SAE, the coactivating proteins have increased sequence similarity compared to non-coactivating proteins (−0.8310, 0.6009, 0.0528) vs. (−1.8392, 0.2276, 0.0276), and the Pearson correlation between the sequence similarity scores for a neuron and the neuron's interpretability score is also weak (0.200, 0.135, 0.142). *SI Appendix*, Table S5 provides this information for all other layers.

*SI Appendix*, section S2 contains some additional tables related to our results. *SI Appendix*, Table S1 shows the percentage of interpretable features in our SAEs and transcoders at various interpretability thresholds. For instance, for layer 9's AA-level SAE: If we employ a threshold of 0.7 for a neuron being considered interpretable, then 50.50% of the randomly analyzed neurons are interpretable; it is 74.00% (threshold 0.5), 62.50% (0.6), 37.50% (0.8) at some other thresholds. In *SI Appendix*, Table S2, we indicate the level of sparsity for neurons in the various SAEs and transcoders at all layers, by providing the mean and median percentages of the number of proteins that activate a sparse neuron. For instance, for the AA-level SAE (layer 9), the mean activation percentage of a neuron is 2.36% of the proteins and the median activation percentage of a neuron is 0.65% of the proteins. While for the protein-level SAE (layer 9), the mean activation percentage of a neuron is 0.32% of the proteins and the median activation percentage of a neuron is 0.02% of the proteins. Interpretations for all sparse features analyzed (across all trained SAEs and transcoders) and ESM neurons, along with their interpretability scores, are available in *SI Appendix*. See *SI Appendix*, section S5 for more details.

## Discussion

Our work extracts interpretable features in multiple ways from protein-level representations using sparse autoencoders and transcoders. Both kinds of representations, derived from PLMs, benefit from interpretability. Protein-level representations are widely used in protein-level downstream tasks, making their interpretability crucial for human–AI collaboration and enhancing trust in downstream bio-AI applications. Extracting interpretable features concealed in a particular representation can help researchers explain and debug downstream performance, and optimize the choice of representation layer for different downstream applications.

On the other hand, interpreting AA-level representations sheds light on the biological features captured by PLMs at each layer. In SAEs trained on both kinds of representations, we identify sparse features that correspond to specific gene ontology terms, protein families, and protein functions. SAEs and transcoders disentangle these features in a completely unsupervised manner, thus providing a means to uncover biologically relevant information. These sparse features could potentially be used to learn more about both well-known and poorly understood protein families and functions, and about how they interact.

Our work highlights both SAEs and transcoders as promising unsupervised approaches for disentangling interpretable features in protein-level representations. As PLMs continue to get larger and better, we expect SAEs and their variants to play a more integral role in uncovering interpretable features present within their representations.

As open-source PLMs become more powerful, it is also vital to evaluate whether they unintentionally encode sensitive or

dangerous biological information that can be used for nefarious purposes such as that which could be used for bioweapons. The ability to interpret and audit PLM representations in an unsupervised manner such as with SAEs and transcoders could help mitigate risks and prevent misuse.

Despite the promising results, our work has certain limitations. For protein-level representations, which are a major focus of our work, we interpret mean-pooled protein-level representations, a popular choice in downstream protein-level tasks (4), but it remains to be seen whether other extraction methods (e.g., beginning-of-sequence (BOS) token representations) yield different sets of interpretable features. Investigating how different representation strategies influence interpretability could be an important direction for future research. We acknowledge that while interpretability methods like SAEs and transcoders hold promise for interpreting features that a PLM has learned, as unsupervised methods, we do not believe these methods are built for predicting protein function. This is because unlike supervised approaches to function prediction, or prediction of any other sort, SAEs and transcoders are unsupervised methods which stand at a significant disadvantage to even the simplest of supervised methods (21).

The type of SAEs we use here for both the protein-level and amino acid-level representations are the TopK SAEs (10), which are a very popular choice for interpreting LLMs; however, there are a few other kinds of SAEs such as those using the L1 norm (9) or Gated SAEs (22). It is possible that some other SAE architecture, for example, the recently proposed BatchTopK (23), may be more suitable depending on the kind of representation with which one works. It has also been documented that features extracted from SAEs, more so TopK SAEs, in the natural language setting can differ depending on the random seed used to initialize weights (24). Although we do not assess this effect in protein foundation models, it is possible that similar variations exist in our extracted features. Understanding how consistent the extracted features are across different random initializations would be valuable to ensure optimal feature extraction.

The automated interpretability protocol we use to quantify the interpretability of sparse features or neurons is meant to serve as a proxy for this task. Although this approach is scalable, we expect it to have limitations, including the effects of both the language used in the interpretation and simulation prompts to the LLM, as well as the type and quality of metadata provided as context. Due to API constraints, we interpret only a subset (200 neurons) of the 20,000 sparse features for each SAE and transcoder. A more exhaustive interpretation would likely reveal additional biologically relevant features and refine our understanding of the representations. It would also enable us to determine the number of proteins for which at least one interpretable sparse feature activates. It would be helpful to develop an autointerpretability protocol that is able to feasibly accommodate AA-level sparse representations without the need for mean-pooling across the amino acids. This would enable us to extract much more granular information from the trained AA-level SAE and pinpoint one specific amino acid at a time (such as catalytic amino acids in enzymes). Circumventing this limitation of our method would likely allow us to find very particular amino acid determinants of specific GO features.

Contemporaneous to our work, Simon and Zou (25) and Adams et al. (26) train and interpret sparse autoencoders on PLM representations, with the former's work and ours leveraging LLM-based interpretability. Although these studies focus exclusively on training and interpreting SAEs on amino acid-level representations, we train and interpret SAEs on both protein-level and amino acid-level representations, as well as transcoders. Protein-level representations, which are a major focus of our work, are widely used for downstream protein prediction tasks (4), making such SAEs and transcoders particularly relevant for interpretability in the context of such downstream applications. Brixi et al. (27) have also contemporaneously trained and interpreted sparse autoencoders on a genomic foundation model (Evo2).

Future work could explore how different methods for extracting protein-level representations impact the kind of sparse features learned by SAEs. If representation choices (e.g., mean pooling vs. BOS token extraction) significantly affect what sparse features are extracted, this could inform best practices for protein-level downstream applications, depending on the task at hand. Investigating how the nature of sparse features extracted varies across different types of SAEs would also be helpful for guiding further use of SAEs in PLMs. We anticipate the extension of similar interpretability methods to other biological foundation models beyond ESM2 will aid in establishing SAEs and their variants as a standard tool for interpreting protein language and other representations in biology.

## Materials and Methods

**SAE: Data.** We start with a dataset of 50 million protein sequences from the Uniref50 dataset (28), while excluding sequences present in the UniProt dataset (17) or sequences longer than 1024 residues. We split the dataset into an 80-20 training-validation split. Then, to get the protein-level representation for each sequence, we mean-pool (after removing the BOS and end-of-sequence (EOS) representations) the token representations extracted from layer L of the ESM2 model: "esm2 t12 35M" UR50D. This dataset is used for training the SAE and transcoder on protein-level representations for layer L. On the other hand, the SAE on amino acid-level representations is trained directly (i.e. without mean-pooling) on the individual (non-BOS and non-EOS) token-level representations for layer L.

**SAE: Model.** Sparse autoencoders can be of different kinds depending on the sparsity constraint imposed on the hidden layer. The two most popular kinds of SAEs are Gao et al.'s (10) TopK SAE [used by Adams et al. (26)] and Bricken et al.'s (9) L1 SAE [used by Simon and Zou (25)]. We work with the TopK SAE, though we do not normalize the decoder. Let the protein-level representation or amino acid-level representation from layer $L$ be $x_L$. Due to the ESM model we work with, $x_L \in R^{480}$. Let $W_{\text{enc}}$ be the encoder weight matrix, $b_{\text{enc}}$ be the encoder bias, $b_{\text{dec}}$ be the decoder bias and $W_{\text{dec}}$ be the decoder weight matrix. Let $x_{L,\text{norm}}$ be the layer normalized version of $x_L$ and let $\text{Top}_k$ be the function that only allows the top $k$ entries to pass through as is, zeroing out the rest. Then the encoder of the sparse autoencoder $S_{\text{enc}}$ is defined as follows:

$$y := S_{\text{enc}}(x_L) = \text{ReLU}\left(\text{Top}_k\left(W_{\text{enc}}(x_{L,\text{norm}} - b_{\text{dec}}) + b_{\text{enc}}\right)\right)$$

We set $k$ to be 64 for the protein-level representations and $k$ to be 16 for the AA-level representations. Now $y$ is in the latent space $R^{20,000}$, which will contain the sparse features (i.e. sparse neurons).

The decoder $S_{\text{dec}}$ is defined as follows:

$$z := S_{\text{dec}}(y) = \text{ULN}\left(W_{\text{dec}}(y) + b_{\text{dec}}\right),$$

where *ULN* simply undoes the normalization that we performed with layernorm in the first step, by multiplying by the SD and adding the mean.

**SAE: Training.** The main loss function for the sparse autoencoder is the standard mean square error reconstruction loss

$$\text{Mean Square Error (MSE)}(a,z) = \frac{1}{n}\sum_{i=1}^{n}(a_i - z_i)^2,$$

where $a = x_L$ for the sparse autoencoder. Though this definition of the sparse autoencoder is sufficient in principle to create a sparse representation of the input $x_L$, it is known to encounter optimization issues (10) due to which a vast majority of neurons in the latent space end up dead (inactive on any inputs). So to solve this, there is another reconstruction term called aux reconstruction (10) that is described along with further training details in *SI Appendix*, section S3.

**Transcoder Details.** For the transcoders, the dataset construction, model architecture, and training method stay the same as above, except for two differences: First, we replace $a = x_L$ with $a = x_{L+1} - x_L$, where $x_{L+1}$ is the representation from the L+1th layer. We take the difference due to skip connections in ESM2 because we want the transcoder to learn to predict only the portion that was newly added across that layer, instead of the combined sum including the residual from the previous layer. Second, since the input and desired output are from different layers in a transcoder, the *ULN* function is no longer employed for transcoders, encouraging the model to directly learn the unnormalized desired output.

**Data for Automated Interpretations Using Claude.** First, we detail the procedure for protein-level representations and then later mention a preprocessing modification necessary when working with the amino acid-level representations. We fix a random seed of 42 and use it to pick 50,000 random protein sequences from UniProt that are no longer than 1024 residues. This dataset of 50,000 random proteins is derived from UniProtKB's reviewed portion: UniProt, a highly curated protein database of over 500,000 protein entries that emphasizes minimal redundancy. In UniProt, sequences from the same gene and species are consolidated into a single entry. This approach ensures that each protein is represented uniquely, minimizing redundancy across the dataset.

We extract the protein-level representations from ESM2 as described before and the sparse representations from our trained SAE or transcoder for these sequences. Then we apply min-max normalization (19) to all sparse features (that are not completely inactive) and retain activation levels up to 10 decimal places for precision in the interpretation step.

For the purposes of comparing the interpretability between sparse features and ESM neurons for protein-level representations, it is desirable that the ESM neurons exhibit both active and inactive states, like the sparse features do. So we set a threshold: 0 is a natural choice, so we set 0 as the threshold drawing on Cunningham et al. (11) and Bills et al. (19), and set the activations of ESM neurons below that threshold to 0. Then we similarly apply min-max normalization to the ESM neurons and retain activation levels up to 10 decimal places for precision in the interpretation step.

First, the collection of 20,000 sparse features is preshuffled using a fixed random seed of 42. For each SAE or transcoder that we worked with, we pick 200 consecutive (out of the 20,000 preshuffled) sparse features that are sufficiently active. We consider a sparse feature to be sufficiently active if it has at least twice as many sequences that it is active for, as we will need for the analysis to follow: We do not want to analyze sparse features that are only just barely active. Drawing from Bricken et al. (9), we employ a variety of activation intervals for the automated interpretability protocol. We start by defining blocks containing activation intervals as follows: inactive block contains the trivial activation interval 0 to 0, low-block contains the activation interval 0 to 0.2, mid-block contains the activation intervals 0.2 to 0.4, 0.4 to 0.6, 0.6 to 0.8, and high-block contains the activation interval 0.8 to 1. For each sparse feature we attempt to interpret, we aim to randomly sample the following number of sequences from each interval in their respective blocks: 6 sequences (high-block), 3 sequences (mid-block), 2 sequences (low-block), and 15 sequences (inactive block). We remark, as noted by Bricken et al. (9), that due to sparsity the inactive block is effectively equivalent to a random block–picking completely random sequences across the dataset–since sparse features are highly unlikely to activate on random sequences.

Due to the large number of sparse features (20,000), our SAEs exhibit high levels of sparsity, especially those trained on protein-level representations. So to avoid having to discard too many such sparse features,[¶] we need to implement a generous spillover mechanism: Whenever there are not sufficient sequences for an activation interval, we fill the void by spilling over to as many other intervals (both above and below) as necessary, always preferring intervals closer to the original interval first. This spillover mechanism either eventually accumulates 17 active sequences and 15 inactive sequences for a sparse feature, or terminates unsuccessfully. [In case a sparse feature does not have at least $2 \times 17 = 34$ active sequences and 15 inactive sequences, we skip it (similarly to ref. 9) and pick the next sparse feature for analysis from our preshuffled collection above. But in either case, we always move on to the next sparse feature in our preshuffled collection above.]

We then randomly regard 8 active and 7 inactive sequences from above as interpretation sequences and 9 active and 8 inactive sequences as simulation sequences for our sparse feature. In similar fashion, we pick 200 random ESM neurons (out of the 480), as well as active and inactive sequences associated with each ESM neuron to interpret, and similarly regard 8 active and 7 inactive sequences as interpretation sequences and 9 active and 8 inactive sequences as simulation sequences for our ESM neuron.

**Preprocessing Modification Necessary for AA-Level Representations.** For the amino acid-level representations, we perform the analogous steps as above, but instead of extracting the mean-pooled ESM2 representations, we extract the amino acid-level representations from ESM2 and the sparse representation for each amino acid in the sequence from the corresponding SAE. We note that a lot of the SAE-related interpretability work on LLMs has focused on interpreting short sentence fragments at a time, with fixed context lengths like 64 tokens (10, 11). However, in comparison, our protein sequences have a much larger number of tokens (amino acids) and different proteins contain vastly different numbers of amino acids. So to maintain feasibility as in the protein-level case, we mean-pool the individual AA-level sparse representations into a single representation, and aim to form interpretations for the aggregated (mean) value of a sparse feature (aggregated over the whole sequence).[#] Mean-pooling helps us interpret the mean activation of a sparse feature and ESM neuron respectively over the whole sequence. After mean-pooling, we proceed identically as we do for the protein-level representation.

**Getting Interpretations from Claude.** As mentioned above, we interpret 200 random sparse features (or ESM neurons) for all our SAEs and transcoders. We carry out the following steps for all the individual neurons (sparse or ESM) that we interpret: We give Claude 3.5 Sonnet (claude-3-5-sonnet-20240620) the following information about the interpretation sequences (both active and inactive sequences): sequence ID,[‖] scaled activation level, protein names, gene names, protein families, (GO) biological processes, (GO) cellular components, and (GO) molecular functions. We aim to assess the interpretability of neurons (sparse or ESM) in the context of this metadata, so we ask Claude to come up with a human-readable interpretation of what the neuron appears to be capturing. Several parts of our prompts to Claude (for getting interpretations and using them to simulate activation levels on the simulation sequences) are adapted from the work of Bills et al. (19) and Paulo et al. (20) in the NLP setting. Following Bills et al. (19), we ask Claude to keep its response to no longer than a single sentence–in order to prevent it from coming up with long lists of all the possible individual examples it has seen. To enable precision, the scaled activation levels Claude is provided with are specified up to 10 decimal places. We ask that the interpretation be human-readable (11), to emphasize that the interpretation is meant for humans, especially since we are dealing with biological data. Our prompt stays the same regardless of whether the neuron in question is a sparse feature or ESM neuron. The prompts are included in *SI Appendix*, section S4.

**Simulating Activations with Claude's Interpretation on the Simulation Sequences.** After getting Claude's interpretation for a neuron (sparse or ESM), in a new prompt, we give Claude 3.5 Sonnet (claude-3-5-sonnet-20240620) all the same information (except activation levels) as above for

[¶] Despite the spillover mechanism, several SAE features for protein-level representations need to be discarded (see neurons attempted/neurons analyzed in Table 2).

[#] Pooling (in particular max-pooling) has been used as an aggregation method for SAE activations in the NLP setting (29). Contemporaneously, in the PLM setting, Adams et al. (26) mean-pool sparse representations (albeit for a slightly different use case), while Simon and Zou (25) employ max-pooling.

[‖] Unique and stable entry identifier.

the simulation sequences (both active and inactive sequences), and ask it to return its prediction for the activation levels. The prompts are included in *SI Appendix*, section S4.

**Computing Interpretability Scores.** For each neuron (sparse or ESM), we compute the interpretability score to be the Pearson correlation between the activation values returned by Claude in the simulation step above and the actual (rescaled) activation values for the simulation sequences, following work in the NLP setting (9, 11, 19). Though we provide Claude with highly explicit instructions not to do so, if it returns the incorrect number of prediction values for a neuron, then that neuron is skipped. It is worth remarking that in the simulation step, Claude is never made aware of the fact that a neuron gets skipped if it returns the incorrect number of predictions. So Claude cannot use this to game its way out of neurons it finds hard to provide suitable predictions for.

**Reproducibility.** To help with reproducibility, all the SAE and transcoder training runs, as well as the autointerpretation protocol use a fixed random seed of 42 everywhere. The random seed of 42 used for our autointerpretability results is entirely arbitrary as 42 is a common arbitrary seed choice in machine learning experiments, and we do not expect the conclusions of our automated interpretability to change based on different random seeds.

**Code and Data.** All code used in the project for training and interpreting the sparse autoencoders and transcoders in the various ways is available on Github at: https://github.com/onkarsg10/Rep_SAEs_PLMs (30). Our final SAE and transcoder training implementation draws on implementation ideas, and various components and modules of code from the following open-source implementations (31–39), along with modifications.

The datasets used for training the SAEs and transcoders is (Uniref50 in .fasta.gz format) (28) which can be downloaded from the provided link. The file was downloaded in .fasta.gz format around Nov, 2024. The dataset used for the GO analysis and interpreting our SAEs is (UniProt in TSV format) (17), which can be downloaded by going to the provided link, selecting the columns you want, and clicking download, while selecting TSV as the file format. For the protein-level representations, we used the file downloaded on Nov 12, 2024, and for the AA-level representations we used the file downloaded on Feb 11, 2025. For the GO analysis we also make use of the go.obo file and the goa_human.gaf file (18, 40) that are downloadable at the indicated links, which we downloaded around Nov, 2024.

**Data, Materials, and Software Availability.** Code data have been deposited in GitHub (https://github.com/onkarsg10/Rep_SAEs_PLMs) (30). Previously published data were used for this work (17, 18, 28, 40).

**ACKNOWLEDGMENTS.** We are extremely thankful to Brian Hie, Bowen Jing, Helen Kang, Shuvom Sadhuka, Sunny Guha, Ali Shehper, and Soojung Yang for helpful conversations. We would like to especially thank Jinyeop Song for helpful discussions that led to pooling the Sparse Autoencoder amino acid-level representations. We are grateful to the reviewers for their helpful comments and suggestions. This work and M.B. are partially supported by NIH 1R35GM141861 (to B.B.) and O.G. by NIH U01 CA250554 (to B.B.).

Author affiliations: [a]Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA 02139; [b]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139; [c]Center for Microbiome Informatics and Therapeutics, Massachusetts Institute of Technology, Cambridge, MA 02139; and [d]Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139

1. T. Bepler, B. Berger, Learning the protein language: Evolution, structure, and function. *Cell Syst.* **12**, 654–669 (2021).
2. Z. Lin *et al*., Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* **379**, 1123–1130 (2023).
3. R. Wu *et al*., High-resolution de novo structure prediction from primary sequence. bioRxiv [Preprint] (2022). https://www.biorxiv.org/content/10.1101/2022.07.21.500999v1 (Accessed 22 July 2022).
4. L. C. Vieira, M. L. Handojo, C. O. Wilke, Scaling down for efficiency: Medium-sized transformer models for protein sequence transfer learning. bioRxiv [Preprint] (2024). https://doi.org/10.1101/2024.11.22.624936 (Accessed 31 November 2024).
5. A. Elnaggar *et al*., Toward understanding the language of life through self-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **44**, 7112–7127 (2022).
6. Z. Zhang *et al*., Protein language models learn evolutionary statistics of interacting sequence motifs. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2406285121 (2024).
7. J. Vig *et al*., Bertology meets biology: Interpreting attention in protein language models. bioRxiv [Preprint] (2020). https://doi.org/10.48550/arXiv.2006.15222 (Accessed 31 November 2024).
8. N. Elhage *et al*., *Toy Models of Superposition* (Transform Circuits, 2022).
9. T. Bricken *et al*., *Towards Monosemanticity: Decomposing Language Models with Dictionary Learning* (Transform Circuits, 2023).
10. L. Gao *et al*., Scaling and evaluating sparse autoencoders. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2406.04093 (Accessed 31 November 2024).
11. H. Cunningham, A. Ewart, L. Riggs, R. Huben, L. Sharkey, Sparse autoencoders find highly interpretable features in language models. arXiv [Preprint] (2023). https://doi.org/10.48550/arXiv.2309.08600 (Accessed 31 November 2024).
12. C. Kissane, R. Krzyzanowski, J. I. Bloom, A. Conmy, N. Nanda, Interpreting attention layer outputs with sparse autoencoders. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2406.17759 (Accessed 31 November 2024).
13. T. Lieberum *et al*., Gemma scope: Open sparse autoencoders everywhere all at once on Gemma 2. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2408.05147 (Accessed 31 November 2024).
14. J. Dunefsky, P. Chlenski, N. Nanda, Transcoders find interpretable LLM feature circuits. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2406.11944 (Accessed 31 November 2024).
15. Z. He *et al*., Llama scope: Extracting millions of features from Llama-3.1-8b with sparse autoencoders. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2410.20526 (Accessed 31 November 2024).
16. M. Chen, J. Engels, M. Tegmark, Low-rank adapting models for sparse autoencoders. arXiv [Preprint] (2025). https://doi.org/10.48550/arXiv.2501.19406 (Accessed 5 February 2025).
17. The UniProt Consortium, Uniprot: The universal protein knowledgebase in 2021. *Nucleic Acids Res.* **49**, D480–D489 (2020).
18. M. Ashburner *et al*., Gene ontology: Tool for the unification of biology. The gene ontology consortium. *Nat. Genet.* **25**, 25–29 (2000).
19. S. Bills *et al*., Language models can explain neurons in language models. OpenAI Public (2023). https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html. Accessed 10 May 2023.
20. G. Paulo, A. Mallen, C. Juang, N. Belrose, Automatically interpreting millions of features in large language models. arXiv [Preprint] (2024). https://arxiv.org/abs/2410.13928 (Accessed 31 December 2024).
21. L. Smith *et al*., Negative results for SAEs on downstream tasks and deprioritising SAE research (GDM mech interp team progress update 2). AI Alignment Forum (2025). https://www.lesswrong.com/posts/4uXCAJNuPKtKBsi28/negative-results-for-saes-on-downstream-tasks. Accessed 28 March 2025.
22. S. Rajamanoharan *et al*., Improving dictionary learning with gated sparse autoencoders. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2404.16014 (Accessed 31 November 2024).
23. B. Bussmann, P. Leask, N. Nanda, Batchtopk sparse autoencoders. arXiv [Preprint] (2024). https://doi.org/10.48550/arXiv.2412.06410 (Accessed 10 December 2024).
24. G. Paulo, N. Belrose, Sparse autoencoders trained on the same data learn different features. arXiv [Preprint] (2025). https://doi.org/10.48550/arXiv.2501.16615 (Accessed 30 January 2025).
25. E. Simon, J. Zou, Interplm: Discovering interpretable features in protein language models via sparse autoencoders. bioRxiv [Preprint] (2024). https://doi.org/10.1101/2024.11.14.623630 (Accessed 15 January 2025).
26. E. Adams, L. Bai, M. Lee, Y. Yu, M. AlQuraishi, From mechanistic interpretability to mechanistic biology: Training, evaluating, and interpreting sparse autoencoders on protein language models. bioRxiv [Preprint] (2025). https://doi.org/10.1101/2025.02.06.636901 (Accessed 20 March 2025).
27. G. Brixi *et al*., Genome modeling and design across all domains of life with Evo 2. bioRxiv [Preprint] (2025). https://doi.org/10.1101/2025.02.18.638918 (Accessed 20 March 2025).
28. B. E. Suzek, H. Huang, P. McGarvey, R. Mazumder, C. H. Wu, Uniref: Comprehensive and non-redundant Uniprot reference clusters. *Bioinformatics* **23**, 1282–1288 (2007).
29. T. Bricken *et al*., *Using Dictionary Learning Features As Classifiers* (Transform Circuits, 2024).
30. O. Gujral, M. Bafna, Rep_SAEs_PLMs. Github. https://github.com/onkarsg10/Rep_SAEs_PLMs. Deposited 28 June 2025.
31. OpenAI, Sparse autoencoder (2024). Github. https://github.com/openai/sparse_autoencoder/tree/4965b941e9eb590b00b253a2c406db1e1b193942. Deposited 30 June 2024.
32. D. Kerrigan, Saefarer (2024). Github. https://github.com/DanielKerrigan/saefarer/tree/41b5c6789952310f515c461804f12e1dfd1fd32d. Deposited 6 September 2024.
33. EleutherAI, SAE (2024). Github. https://github.com/EleutherAI/sae/tree/main. Deposited 13 November 2024.
34. B. Bussmann, Batchtopk (2024). Github. https://github.com/bartbussmann/BatchTopK/tree/main. Deposited 2 September 2024.
35. T. Cosgrove, Sparse-autoencoder-mistral7b (2024). Github. https://github.com/tylercosgrove/sparse-autoencoder-mistral7b. Deposited 7 August 2024.
36. C. T. Joseph Bloom, D. Chanin, Saelens (2024). Github. https://github.com/jbloomAus/SAELens. Deposited 29 December 2024.
37. N. Nanda, 1l-sparse-autoencoder (2023). Github. https://github.com/neelnanda-io/1L-Sparse-Autoencoder. Deposited 28 October 2023.
38. E. Adams, interprot (2024). Github. https://github.com/etowahadams/interprot/tree/cbe293403e6427b3a7891f8dde15d2f8d7f81d96/interprot. Deposited 5 November 2024.
39. T. Lawson, MLSAE (2024). Github. https://github.com/tim-lawson/mlsae/tree/main/mlsae/model. Deposited 20 December 2024.
40. Gene Ontology Consortium *et al*. The gene ontology knowledgebase in 2023. *Genetics* **224**, iyad031 (2023).