PairSAE: Mechanistic Interpretability from Pair Representations in Protein Co-Folding

Giosue Migliorini[*]
University of California, Irvine

Aristofanis Rontogiannis
Flagship Pioneering

Grigori Guitchounts
Flagship Pioneering

Nicholas Franklin
Flagship Pioneering

Axel Elaldi
Flagship Pioneering

Olivia Viessmann
Flagship Pioneering

Abstract

Foundation models for structural biology have achieved remarkable performance in predicting biomolecular structure and show promise for the design of proteins and small molecules. Yet understanding *which* internal features drive their outputs remains challenging. Standard sparse autoencoders (SAEs), effective on transformer-style *sequence embeddings*, do not transfer cleanly to pairformer-like architectures: naïvely operating on pairwise representations yields a quadratic blow-up of features and obscures concepts distributed jointly across sequence and pair representations. We introduce *PairSAE*, which summarizes pairwise tensors via an N-mode SVD into token-wise interaction roles, then uses a sparse autoencoder to learn a *shared* set of token-level features that decode into both sequence and pair representations. Evaluated on Boltz-2 activations for PLINDER protein–ligand complexes, PairSAE yields interpretable features that align with UniProt annotations and predict Boltz-2 affinity values. These results indicate that PairSAE links the latent space of foundation models for structural biology to interpretable structural concepts, clarifying what the model "knows" while avoiding pairformer-induced pitfalls that limit conventional SAEs.

1 Introduction

Recent advances in protein structure prediction, most notably AlphaFold and the Boltz models [Jumper et al., 2021, Abramson et al., 2024, Wohlwend et al., 2025, Passaro et al., 2025], have transformed structural biology and now extend to DNA, RNA, small molecules, and their complexes. These models achieve high accuracy without explicitly encoding physical or chemical laws, instead exploiting statistical regularities in large datasets of experimentally determined structures. This raises a central question: to what extent do they implicitly capture the physics and chemistry that govern folding? Answering this requires interpretability – explanations that let researchers assess prediction reliability and biophysical plausibility – yet the highly nonlinear architectures of deep models make such analysis challenging. Work in mechanistic interpretability shows that apparent "polysemantic" neurons can arise when networks represent more sparse features than available dimensions, forcing features into superposition rather than cleanly localizing them [Elhage et al., 2022, Scherlis et al., 2022]. Sparse autoencoders (SAEs) address this by disentangling superposed features into more monosemantic components via sparse, overcomplete dictionaries [Olshausen and Field, 1997, Ng et al., 2011, Elhage et al., 2022]. Results in large language models [Cunningham et al., 2024, Bricken et al., 2023] and protein language models (pLMs) [Simon and Zou, 2024, Adams et al., 2025] suggest that SAEs can recover interpretable latent features, making them a suitable candidate for probing the representations learned by protein structure prediction models

[*]Work done during an internship at Flagship Pioneering.

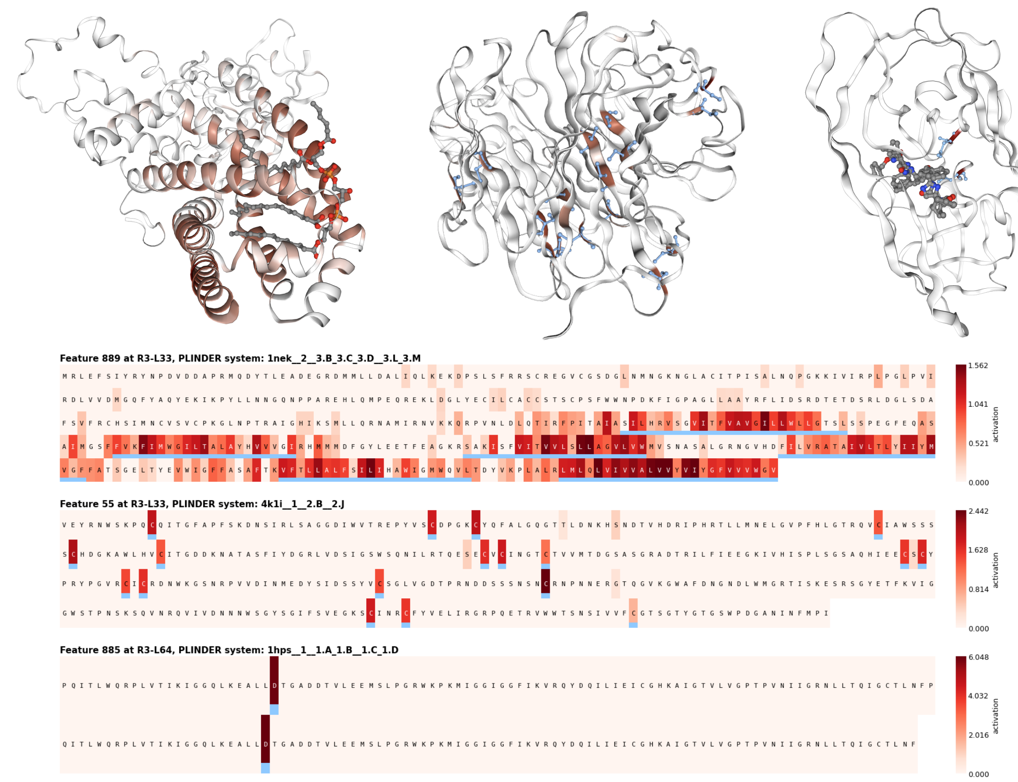

Figure 1: Interpretable features recovered by PairSAEs trained at layers 33 and 64 (third recycling step) of Boltz-2. **Top left:** feature 889 on *E. coli* Complex II (PDB 1NEK); activates on ***trans-membrane*** proteins (token $F_1 = 0.58$, complex $F_1 = 0.65$). **Top center:** feature 55 on influenza N2 neuraminidase chain B (PDB 4K1I); activates on ***disulfide bonds*** (token $F_1 = 0.79$, complex $F_1 = 0.81$). **Top right:** feature 885 on an HIV-1 protease inhibitor (PDB 1HPS); activates on ***protease active sites*** shared with a dimeric partner (token $F_1 = 0.93$, complex $F_1 = 0.93$). **Bottom:** per-residue activations with UniProt ground-truth (blue).

However, applying SAEs naively to structure prediction models is nontrivial. Unlike most transformer-based LLMs and pLMs, these models use pairformer blocks with pairwise representations alongside sequence embeddings [Abramson et al., 2024]. Learning separate dictionaries per pair scales quadratically and obscures analysis, and repeated pair–sequence interactions suggest features may be superposed across both spaces. To address this, we introduce *PairSAE*, which reconstructs both sequence-level and pairwise embeddings from a shared feature set. Applied to Boltz-2, PairSAE links latent representations to interpretable structural concepts, clarifying what the model "knows" about folding and enabling more principled evaluation and discovery in structural biology.

2 Background

2.1 Sparse autoencoders

Sparse dictionary learning [Olshausen and Field, 1997, Lee et al., 2006, Ng et al., 2011] is a long-standing representation learning technique that aims at finding a dictionary $\mathbf{D} = \{\mathbf{d}_j\}_{j=1}^{D}$ approximating input vectors $\mathbf{x} \in \mathbb{R}^n$ via a linear combination of its columns with a set of sparse latent features $\mathbf{h} \in \mathbb{R}^D$, extracted from $\mathbf{x}$ itself. It has recently been shown that this formulation can yield interpretable features in transformed-based models such as large language models [Elhage et al., 2022, Templeton et al., 2024, Cunningham et al., 2024], protein language models like ESM2 [Simon and Zou, 2024, Garcia and Ansuini, 2025, Adams et al., 2025, Gujral et al., 2025, Parsan et al., 2025], and foundation models for genomics like Evo 2 [Brixi et al., 2025].

Such features can be obtained by training a sparse autoencoder using a reconstruction loss with a sparsity constraint. In mech interp the autoencoder is typically linear, with both a sparsity and positivity constraint on the latent features. A common construction is

$$\mathbf{h} = \sigma\left(\mathbf{E}\,\mathbf{x} + \mathbf{b}_{\text{enc}}\right), \quad \hat{\mathbf{x}} \;=\; \mathbf{D}\,\mathbf{h} \;+\; \mathbf{b}_{\text{dec}}, \tag{1}$$

where $\mathbf{E} \in \mathbb{R}^{D\times n}$, $\mathbf{b}_{\text{enc}} \in \mathbb{R}^D$ are the weights and bias terms for the encoder, $\mathbf{D} \in \mathbb{R}^{n\times D}$, $\mathbf{b}_{\text{dec}} \in \mathbb{R}^n$ are the weights and bias terms for the decoder, and σ is a sparsity-inducing nonlinearity such as ReLU [Cunningham et al., 2024, Bricken et al., 2023], TopK [Gao et al., 2025], BatchTopK [Bussmann et al., 2024], and others.

2.2 Pairformer representations

In a pairformer module [Abramson et al., 2024], an input sequence $\mathbf{x}$ of length N_{tok} tokens is encoded into a sequence embedding matrix $\mathbf{S} \in \mathbb{R}^{N_{\text{tok}}\times n_s}$, whose i-th row is the token embedding vector $\mathbf{s}_i \in \mathbb{R}^{n_s}$, and a three-dimensional pair representation tensor $\mathcal{Z} \in \mathbb{R}^{N_{\text{tok}}\times N_{\text{tok}}\times n_z}$, whose (i, j)-th slice $\mathcal{Z}_{i,j} \in \mathbb{R}^{n_z}$ represents the embedding of token pair (i, j). Here, n_s and n_z are the sequence-level and pairwise embedding dimensions, respectively. Pair representations control the information flow in the updates of the sequence-level embeddings at each layer of the pairformer, by biasing the attention logits Abramson et al. [2024]. Pair representations were first introduced with the Evoformer, part of the architecture of AlphaFold 2 Jumper et al. [2021].

3 PairSAE

Motivation. Our goal is to obtain token-level sparse features that jointly reconstruct sequence and pair representations. Pairwise embeddings impose no structure, so we first compress them into a token-wise summary that preserves row/column interaction roles. A shared feature set then decodes into both representation types.

Summarizing pairwise interactions. We propose to perform an $N-$mode singular value decomposition (SVD) of the tensor $\mathcal{Z}$, and use it to construct a sequence representation that is specifically designed to incorporate information about how each token interacts in the sequence. We include a brief introduction to $N-$mode SVD in Appendix A. Information about the role that each token plays *as a row and as a column* in $\mathcal{Z}$ can be obtained by inspecting the mode-1 and mode-2 matrices of left singular vectors $\mathbf{U}^{(1)}, \mathbf{U}^{(2)} \in \mathbb{R}^{N_{\text{tok}}\times N_{\text{tok}}}$ [De Lathauwer et al., 2000a, Vasilescu and Terzopoulos, 2002]. We consider a truncation to the first r columns $\mathbf{U}^{(1)}_{:,1:r}$ and $\mathbf{U}^{(2)}_{:,1:r}$, ordered by their singular value. We then concatenate them column-wise into a new sequence level embedding

$$\mathbf{m}_i = \left[\mathbf{U}^{(1)}_{i,1:r} \parallel \mathbf{U}^{(2)}_{i,1:r}\right], \quad i = 1, \dots, N_{\text{tok}}. \tag{2}$$

Autoencoder. PairSAE latent features are obtained by encoding a concatenation of this new embedding matrix with the original sequence-level representations from the pairformer model:

$$\mathbf{h}_i = \sigma\left(\mathbf{E}\left[\mathbf{s}_i \parallel \mathbf{m}_i\right] + \mathbf{b}_{\text{enc}}\right),, \quad i = 1, \dots, N_{\text{tok}}, \tag{3}$$

where $\mathbf{E} \in \mathbb{R}^{D\times(n_s+2r)}$, $\mathbf{b}_{\text{enc}} \in \mathbb{R}^D$, and σ is a composition of BatchTopK and ReLU [Bussmann et al., 2024]. We decode with shared features into both spaces:

$$\hat{\mathbf{s}}_i = \mathbf{D}^s\,\mathbf{h}_i + \mathbf{b}^s_{\text{dec}}, \quad \hat{\mathbf{z}}_{i,j} = \mathbf{D}^{z_{\text{row}}}\mathbf{h}_i + \mathbf{D}^{z_{\text{col}}}\mathbf{h}_j + \mathbf{b}^z_{\text{dec}}, \quad i, j \in \{1, \dots, N_{\text{tok}}\}, \tag{4}$$

where $\mathbf{D}^s \in \mathbb{R}^{n_s\times D}$, $\mathbf{b}^s_{\text{dec}} \in \mathbb{R}^{n_s}$, $\mathbf{D}^{z_{\text{row}}}, \mathbf{D}^{z_{\text{col}}} \in \mathbb{R}^{n_z\times D}$, and $\mathbf{b}^z_{\text{dec}} \in \mathbb{R}^{n_z}$.

Loss function. Choosing the dictionary size D can have major consequences on the types of features that are learned, and a larger dictionary does not necessarily translate to better performance on downstream tasks [Templeton et al., 2024, Gao et al., 2025]. It has recently been shown that a Matryoshka SAE loss [Bussmann et al., 2025] exhibits robust performance across dictionary sizes. This is attained by slicing the decoder matrix and the feature vector across several nested levels (we

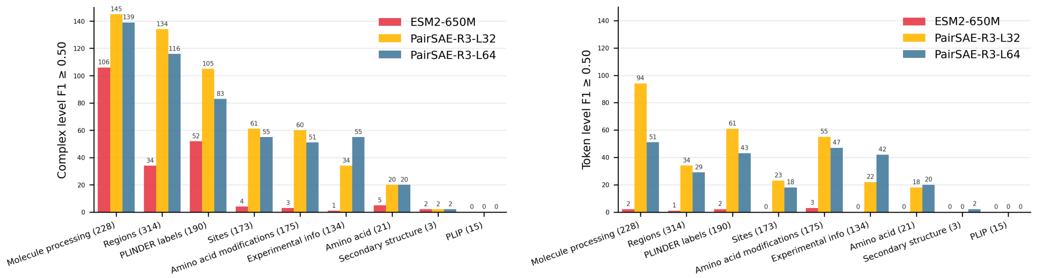

Figure 2: Count of concepts with $F_1 \geq 0.5$ using complex-level recall (left) and token level (right), grouped by UniProt annotation category (test-set counts in parentheses).

use three nested widths $c_1 < c_2 < c_3 = D$), and summing them to compute the objectives

$$\mathcal{L}(\mathbf{s}_i) \coloneqq \sum_{k=1}^{3} \left\| \mathbf{s}_i - \mathbf{D}^{s}_{[:,\,1:c_k]} \mathbf{h}_{i,[1:c_k]} - \mathbf{b}^{s}_{\text{dec}} \right\|_2^2, \tag{5}$$

$$\mathcal{L}(\mathbf{z}_{ij}) \coloneqq \sum_{k=1}^{3} \left\| \mathbf{z}_{ij} - \mathbf{D}^{z_{\text{row}}}_{[:,\,1:c_k]} \mathbf{h}_{i,[1:c_k]} - \mathbf{D}^{z_{\text{col}}}_{[:,\,1:c_k]} \mathbf{h}_{j,[1:c_k]} - \mathbf{b}^{z}_{\text{dec}} \right\|_2^2 \tag{6}$$

Following [Gao et al., 2025], we include an auxiliary loss function $\mathcal{L}_{\text{aux}}$ that can "revitalize" dead features, resulting in the loss

$$\mathcal{L}(\mathbf{S}, \mathcal{Z}) = \sum_{i=1}^{N_{\text{tok}}} \mathcal{L}(\mathbf{s}_i) + \lambda_{\text{pair}} \sum_{j=1}^{N_{\text{tok}}} \mathcal{L}(\mathbf{z}_{ij}) + \lambda_{\text{aux}} \mathcal{L}_{\text{aux}}. \tag{7}$$

We speed up training by we computing a Monte Carlo approximation of the second summation and only consider a single j for each i. We set $\lambda_{\text{pair}} = N_{\text{tok}}^{-1}$ to have both representation be equally weighted.

4 Results

We evaluate PairSAE on Boltz-2 representations for protein–ligand complexes from PLINDER [Durairaj et al., 2024]. We train two models at the third recycling step, at layers 33 and 64 (R3-L33, R3-L64). Interpretability can be assessed via linear probes [Gao et al., 2025, Simon and Zou, 2024], and we test whether features predict UniProt residue annotations [UniProt, 2025] and PLINDER system annotations.

Method	Token F_1	Complex F_1
ESM2-650M	0.8%	19.7%
PairSAE-R3-L32	29.1%	53.2%
PairSAE-R3-L64	24.0%	49.6%

Table 1: Percentage of concepts in the test set with $F_1 \geq 0.5$.

Boltz-2 demonstrated a strong improvement on the speed/accuracy Pareto frontier for binding affinity prediction [Passaro et al., 2025], and has the potential of further accelerating virtual screenings. We aim to identify features that exhibit strong association with the predicted affinity, and to do so we build on the hypothesis generation techniques laid out in Movva et al. [2025]. Details of each experiment can be found in Appendix B.

4.1 Linear probing

Following [Simon and Zou, 2024], we normalize each feature to $[0, 1]$, fit a single-threshold classifier per concept–feature pair via a grid search with candidate thresholds spaced at increments of 0.1 across the unit interval, and select the best feature by validation F_1. As noted in Simon and Zou [2024], it is often the case that latent features at a single token might capture properties of the entire system, hence we follow Simon and Zou [2024] and compute the F_1 score by considering the recall at the complex level as well as the token level, resulting in two separate metrics. For each metric, we

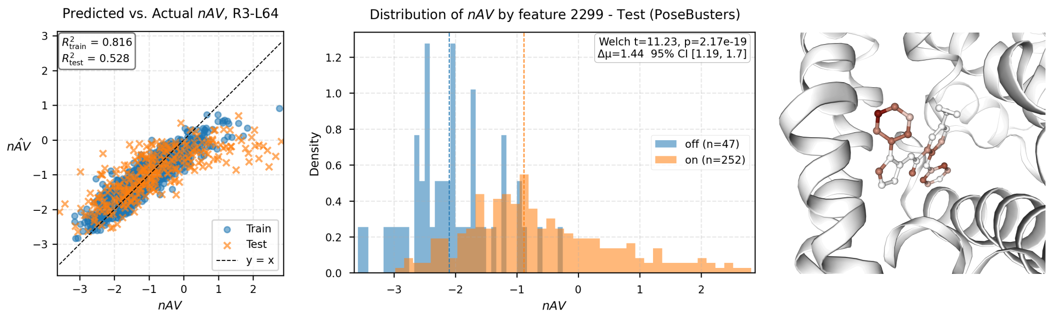

Figure 3: **Left:** negative affinity values from Boltz-2 (higher means more affine) [Passaro et al., 2025], and predicted values from the LASSO regression in (8). **Center:** feature 2299 from R3-L64, displaying strong group difference in affinity values measured by Welch t-test. **Right:** values of feature 2299 overlaid on a ligand where it is highly activated.

display a count of how many features we could predict with a score above 0.5 on a test set in Figure 2. We find that PairSAE features are highly predictive of UniProt annotations, as shown in Table 1, and outperform classifiers based on neurons from the last layer of ESM-2-650M, considered powerful embeddings for many downstream applications [Lin et al., 2023].

4.2 Hypothesis generation: explaining binding affinity predictions

Because affinity is defined at the complex level, we form a complex embedding by max-pooling features over tokens: $\hat{h}_k = \max_{i \le N_{\text{tok}}} h_{i,k}$, $k = 1, \dots, D$. We then predict AV via LASSO [Tibshirani, 1996, Movva et al., 2025], minimizing on a training set $\mathcal{D}$ of 980 PLINDER complexes:

$$\sum_{j \in \mathcal{D}} \left\| AV_j - \boldsymbol{\beta}\, \hat{\mathbf{h}}_j \right\|_2^2 + \lambda \left| \boldsymbol{\beta} \right|_1 . \tag{8}$$

We test our results on Posebusters [Buttenschoen et al., 2024].

Our first finding is on the predictive power of the PairSAE features on affinity values: by choosing λ using cross-validation, we can predict AV with an R^2 of 0.528 from R3-L64 using 291 features out of 16,384. Errors increase at higher predicted affinity, partly due to label shift (fewer high-affinity examples in training than in PoseBusters; see Fig. 3).

Moreover, by letting λ increase we can highlight features that carry most of the predictive power. By looking at the top 10 most influential features at both layers we find most of them to be activated on ligands, for which we have very limited annotations. In Figure 3 we highlight a feature from R3-L64, selected by setting λ in (8) such that only one feature has nonzero coefficient. This feature activates on complexes with higher binding affinity, with a strong group difference when compared to inactive complexes. Additional results are reported in Appendix C.

5 Conclusion

We introduced PairSAE, a sparse dictionary learning method that discovers a shared feature basis jointly explaining sequence and pair representations in pairformer-style models. We showed that these features encode biophysically meaningful concepts and can surface hypotheses about a model's inner workings for downstream tasks such as binding-affinity prediction. Future work includes scaling up the study to more layers and recycling steps, integrating PairSAE features into automated interpretability pipelines to accelerate concept discovery, and developing steering methods for interpretable protein design via feature interventions. A key limitation is that we did not map ligand-activating sparse features to specific concepts, due to the lack of ligand annotations.

References

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *nature*, 596(7873):583–589, 2021.

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

Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Noah Getz, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Liam Atkinson, Tally Portnoi, Itamar Chinn, et al. Boltz-1 democratizing biomolecular interaction modeling. *BioRxiv*, pages 2024–11, 2025.

Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, et al. Boltz-2: Towards accurate and efficient binding affinity prediction. *BioRxiv*, pages 2025–06, 2025.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Adam Scherlis, Kshitij Sachan, Adam S Jermyn, Joe Benton, and Buck Shlegeris. Polysemanticity and capacity in neural networks. *arXiv preprint arXiv:2210.01892*, 2022.

Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *ICLR*, 2024.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. `https://transformer-circuits.pub/2023/monosemantic-features/index.html`, 2023. Transformer Circuits Thread.

Elana Simon and James Zou. Interplm: Discovering interpretable features in protein language models via sparse autoencoders. *bioRxiv*, pages 2024–11, 2024.

Etowah Adams, Liam Bai, Minji Lee, Yiyang Yu, and Mohammed AlQuraishi. From mechanistic interpretability to mechanistic biology: Training, evaluating, and interpreting sparse autoencoders on protein language models. *Forty-second International Conference on Machine Learning*, 2025.

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Ng. Efficient sparse coding algorithms. *Advances in neural information processing systems*, 19, 2006.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL `https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html`.

Edith Natalia Villegas Garcia and Alessio Ansuini. Interpreting and steering protein language models through sparse autoencoders. *arXiv preprint arXiv:2502.09135*, 2025.

Onkar Gujral, Mihir Bafna, Eric Alm, and Bonnie Berger. Sparse autoencoders uncover biologically interpretable features in protein language model representations. *Proceedings of the National Academy of Sciences*, 122(34):e2506316122, 2025.

Nithin Parsan, David J Yang, and John Jingxuan Yang. Towards interpretable protein structure prediction with sparse autoencoders. In *Learning Meaningful Representations of Life (LMRL) Workshop at ICLR 2025*, 2025.

Garyk Brixi, Matthew G Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang, Gabriel A Gonzalez, Samuel H King, David B Li, Aditi T Merchant, et al. Genome modeling and design across all domains of life with evo 2. *BioRxiv*, pages 2025–02, 2025.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000a.

M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *European conference on computer vision*, pages 447–460. Springer, 2002.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025.

Janani Durairaj, Yusuf Adeshina, Zhonglin Cao, Xuejin Zhang, Vladas Oleinikovas, Thomas Duignan, Zachary McClure, Xavier Robin, Gabriel Studer, Daniel Kovtun, et al. Plinder: The protein-ligand interactions dataset and evaluation resource. *bioRxiv*, pages 2024–07, 2024.

UniProt. Uniprot: the universal protein knowledgebase in 2025. *Nucleic acids research*, 53(D1): D609–D617, 2025.

Rajiv Movva, Kenny Peng, Nikhil Garg, Jon Kleinberg, and Emma Pierson. Sparse autoencoders for hypothesis generation. In *Forty-second International Conference on Machine Learning*, 2025.

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

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

Martin Buttenschoen, Garrett M Morris, and Charlotte M Deane. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 15(9):3130–3139, 2024.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. On the best rank-1 and rank-($r_1, r_2, ..., r_n$) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, 21(4): 1324–1342, 2000b.

Nick Vannieuwenhoven, Raf Vandebril, and Karl Meerbergen. A new truncation strategy for the higher-order singular value decomposition. *SIAM Journal on Scientific Computing*, 34(2):A1027–A1052, 2012.

Mariya Ishteva, P-A Absil, Sabine Van Huffel, and Lieven De Lathauwer. Tucker compression and local optima. *Chemometrics and Intelligent Laboratory Systems*, 106(1):57–64, 2011.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Sameer Velankar, José M Dana, Julius Jacobsen, Glen Van Ginkel, Paul J Gane, Jie Luo, Thomas J Oldfield, Claire O'Donovan, Maria-Jesus Martin, and Gerard J Kleywegt. Sifts: structure integration with function, taxonomy and sequences resource. *Nucleic acids research*, 41(D1):D483–D489, 2012.

Sebastian Salentin, Sven Schreiber, V Joachim Haupt, Melissa F Adasme, and Michael Schroeder. Plip: fully automated protein–ligand interaction profiler. *Nucleic acids research*, 43(W1):W443–W447, 2015.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

A N-mode Singular Value Decomposition

The mode-k unfolding of a tensor $\mathcal{T} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_N}$ is obtained by flattening all but the k^{th} dimension into a matrix $\mathcal{T}^{(k)} \in \mathbb{R}^{n_k \times \prod_{i \neq k} n_i}$. Following De Lathauwer et al. [2000a], every tensor admits the higher-order singular value decomposition

$$\mathcal{T} = \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)},$$

where $\mathcal{C} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_N}$ is a core tensor, $\times_k$ denotes the mode-k tensor–matrix product, and each $\mathbf{U}^{(k)} \in \mathbb{R}^{n_k \times n_k}$ is an orthogonal matrix obtained from the left-singular vectors of the mode-k unfolding $\mathcal{T}^{(k)}$ [Vasilescu and Terzopoulos, 2002]. We refer to $\mathbf{U}^{(k)}$ as the mode-k left singular vectors of $\mathcal{T}$.

Unlike matrices, truncating the ordered left-singular vectors to their $r_k < n_k$ columns and fitting a new, smaller core tensor does not yield an optimal $(r_1, r_2, \dots, r_N)$-rank approximation of $\mathcal{T}$ in the Frobenius norm [De Lathauwer et al., 2000b]. Multiple algorithms have been proposed to improve upon a naive truncation [De Lathauwer et al., 2000b, Vannieuwenhoven et al., 2012], but convergence guarantees are limited to local optima [Ishteva et al., 2011].

B Experimental Details

B.1 PairSAE

Following the standard implementation of Boltz-2, our sequence representation has dimension $n_s = 384$, while the pair representation has dimension $n_z = 128$. We compute the $N-$mode SVD by simply flattening $\mathcal{Z}$ to its mode-1 and mode-2 unfolding, and obtain the SVD of these matrices using `numpy.linalg.svd`. We truncate to the first $r = 64$ columns, and if $N_{\text{tok}} < r$ we fill the remaining entries with zeroes. After concatenating sequence embeddings $\mathbf{s}$ and the SVD-derived embedding $\mathbf{m}$ into a 512-dimensional vector, we perform a layer normalization.

We let PairSAE have a dictionary size of $D = 16,384$, corresponding to an expansion factor of 32$\times$. We train on minibatches of 2,048 tokens using the Adam optimizer [Kingma and Ba, 2014], with learning rate 0.0002. We train for 250,000 steps on 40,000 systems sampled from the PLINDER training set [Durairaj et al., 2024] with at most 512 residues, and at each step we sample tokens at random. Due to compute constraints, we do not use the MSA when training and evaluating the PairSAE.

B.2 Linear probing

We consider annotations from UniProt [UniProt, 2025], and binarize them by one-hot encoding each annotation. We map each chain in a PLINDER system to its UniProt annotation using the SIFTS database [Velankar et al., 2012], and match sequences with a minimum of 95% correspondance between the two databases, ignoring sequences that can't be matched.

We also consider a subset of the system-level annotations found in PLINDER, as well as the PLIP interaction fingerprints [Salentin et al., 2015] that are also found in PLINDER. Iterating over features h_i for $i = 1, \dots, D$ and concept annotations y_j for $j = 1, \dots, N_y$, we consider a simple classifier

$$\hat{y}_j = \mathbf{1}\{h_i > \tau_{ij}\}. \tag{9}$$

We pick the best threshold τ_{ij} on a training set of 7,680 complexes from the PLINDER training set, we select the best feature i for each concept j based on F_1 scores on a validation set of 2,560 complexes, and finally report F_1 scores on a test set comprised of 5,120 randomly selected complexes. We restrict our analysis to concepts that appear on at least five different complexes.

B.3 Hypothesis generation

We build a training set for LASSO regression by obtaining Boltz-2 predictions for affinity values in 980 systems from PLINDER, held out from the PairSAE training set. The test set is Posebusters [Buttenschoen et al., 2024]. In figure B.4 we highlight how these differ in terms of the response variable, and note the lack of high affinity complexes in our training set.

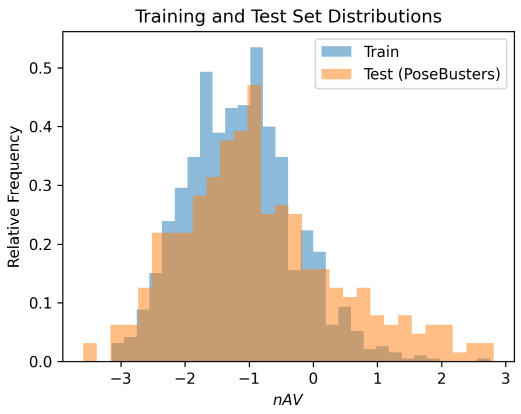

Figure B.4: Boltz-2 affinity values for the training and test set used in the hypothesis generation task.

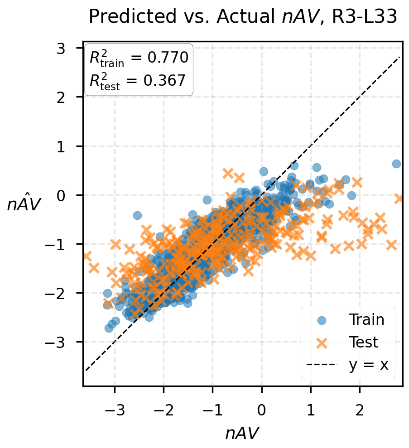

Figure B.5: Negative affinity values from Boltz-2 (higher means more affine), and predicted values from the LASSO regression in (8) using the PairSAE at R3-L33.

LASSO regression is fit using `sklearn.linear_model.Lasso` [Pedregosa et al., 2011]. Detailed results for both models are presented in Table B.2. See Figure B.5 for true vs predicted Boltz-2 affinity values based on the PairSAE at R3-L33.

As mentioned in Section B, we did not use the MSA for our analysis. In Figure B.6 we compare the Boltz-2 output with and without MSA, and note a substantial difference in predicted affinity values and affinity probability. As MSA-based predictions are conditioned on more information, we expect these to be more accurate. In future work, we aim at replicating our analysis using MSA inputs.

Table B.2: LASSO regression on max-pooled PairSAE features. ρ denotes Spearman rank correlation.

		Prediction metrics					**Ranking**		**Sparsity**
Model	λ (CV)	Train R^2	Train MAE	Train RMSE	Test R^2	Test MAE	Train ρ	Test ρ	Nonzero
R3–L64	0.009	0.816	0.275	0.131	0.528	0.607	0.913	0.805	291
R3–L33	0.006	0.770	0.302	0.164	0.367	0.713	0.893	0.706	237

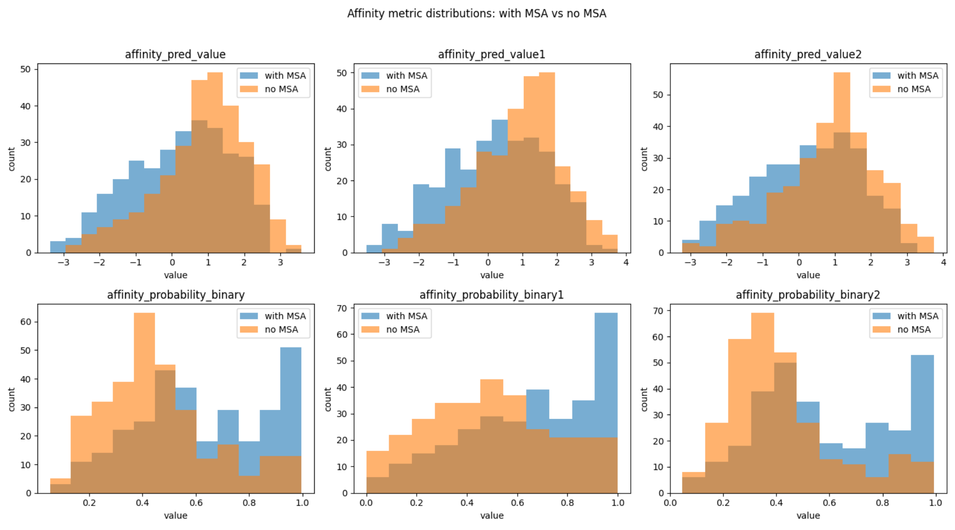

Figure B.6: Comparison of having MSA on and off in the Boltz-2 affinity values and affinity probabilities, in the Posebusters dataset.

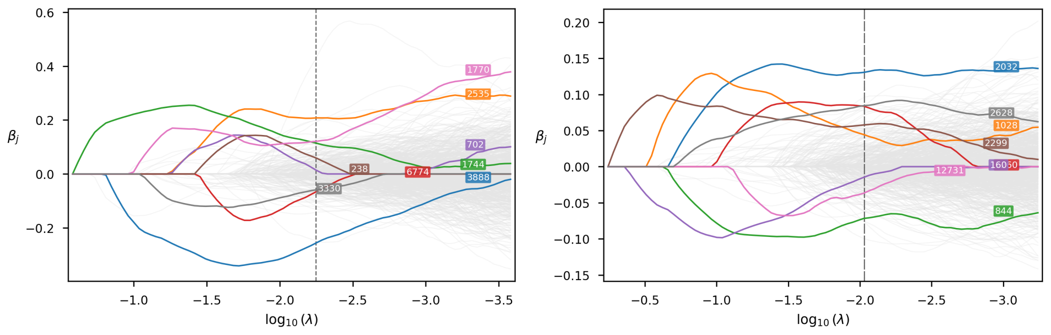

Figure C.7: LASSO regression coefficients for varying values of the λ regularization parameter in (8), fit on the PairSAE at R3-L33 (left) and R3-L64 (right).

C Additional results

In Figure C.7 we display how lasso coefficients evolve for varying levels of regularization, and highlight features having strong association with the predicted affinity. Among these, we highlight how feature 1744 and 1770 from R3-L33 activate in two ligands in Figure C.8.

In Section 4.2 we highlighted feature 2299 from R3-L64, that exhibits a strong group difference by activating on complexes with higher predicted affinity. In R3-L33 we found a feature representing the opposite, and activating only samples that on average have a lower predicted affinity. Group differences for these two features are presented in Figure C.9.

In Figure C.10 we display values of feature 3888 overlaid on ligands where it is activated. In these examples, the predicted complexes do not appear to present a plausible docked pose: the small molecule is displaced from the protein interface and does not form stable contacts.

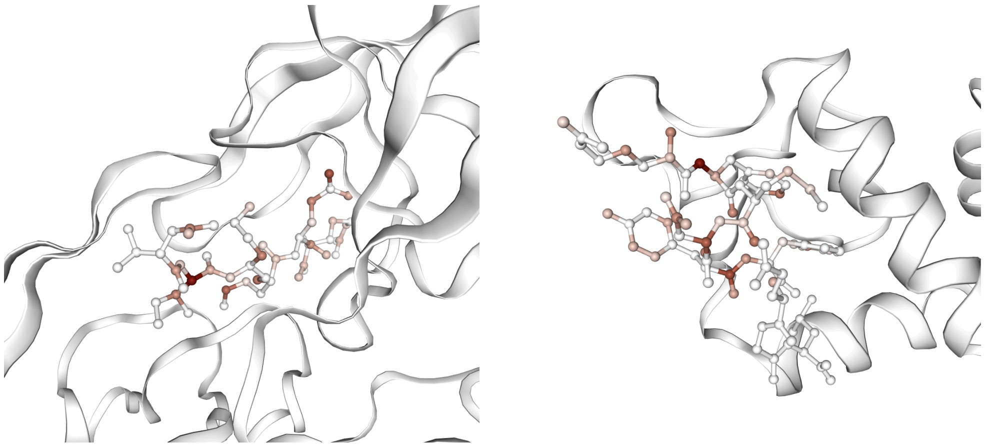

Figure C.8: Ligands activating on feature 1770 (left) and feature 1774 (right). These are the top 2 positive features that predict the Boltz-2 affinity in R3-L33, as highlighted in Figure C.7.

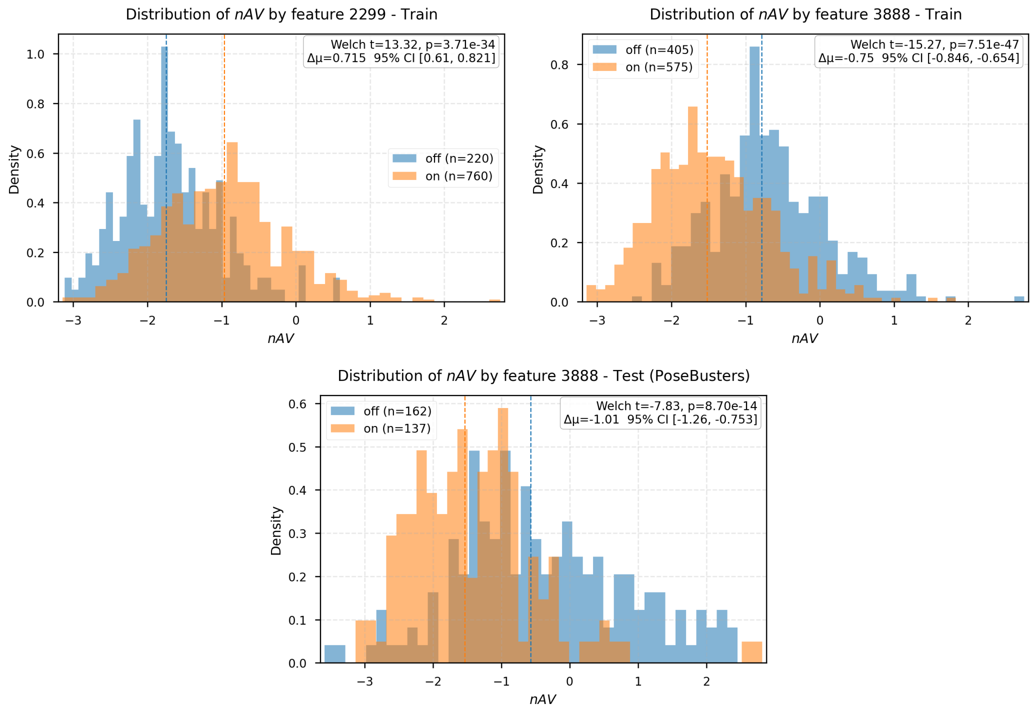

Figure C.9: Group difference in Boltz-2 affinity values when splitting the dataset based on whether a feature activates or not. **Left:** feature 2299 from R3-L64, training set. **Right:** feature 3888 from R3-L33, training set. **Bottom:** feature 3888 from R3-L33, test set.

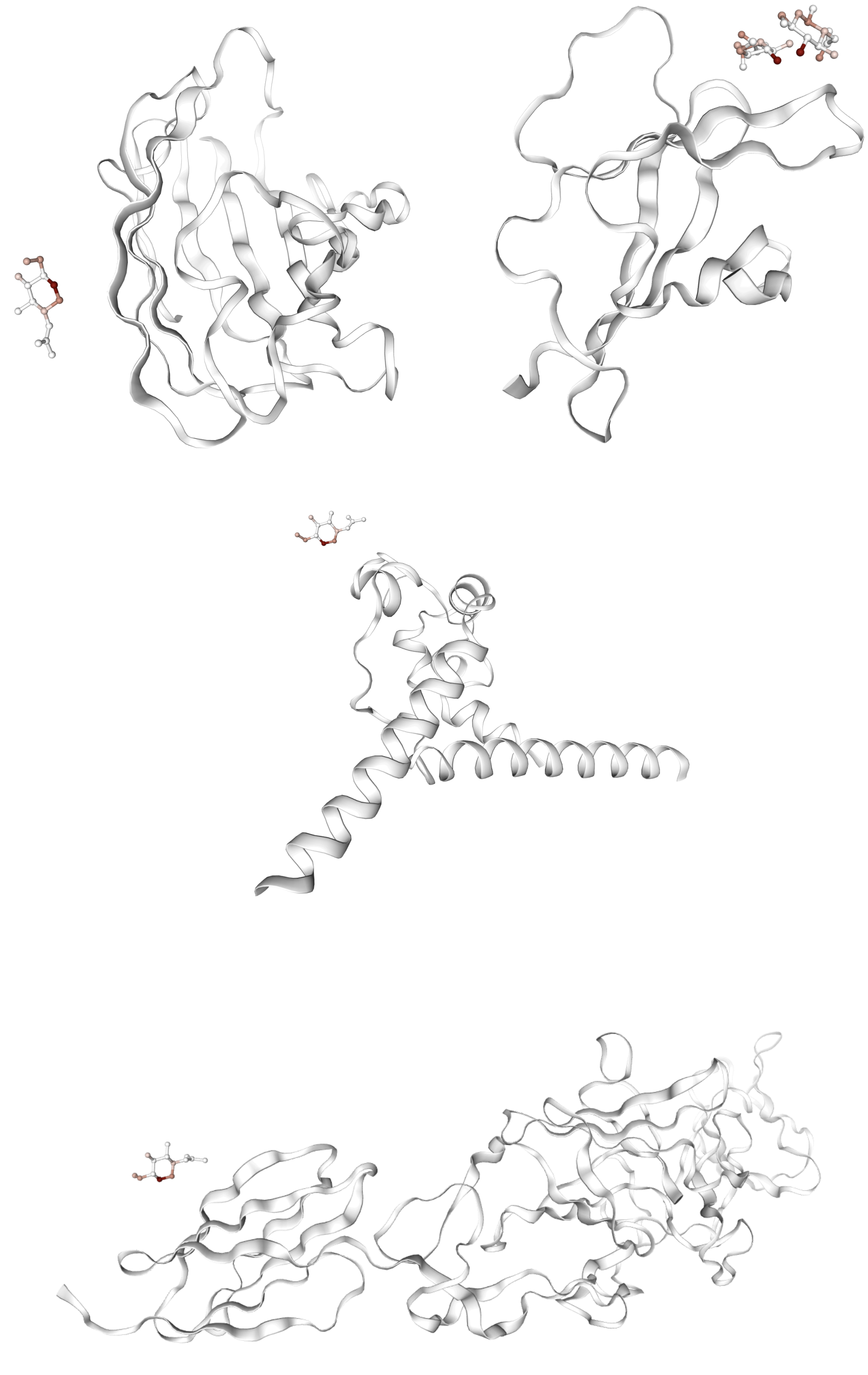

Figure C.10: Examples where feature 3888 is activated.